MobiTicket: une application mobile de ventes aux enchères de billets de spectacles

Présenté par Christina Braz
Centre interuniversitaire de recherche en analyse des organisations (CIRANO)

Prof. Dr. Jacques Robert,
Vice-président groupe commerce électronique
CIRANO

M. Alain Bernardi,
Directeur général, Innovations technologiques
Laboratoires Universitaires Bell (Bell Canada)

Rapport émis en : 27 octobre, 2003

Table de matières

1. Introduction

Le projet d'enchère mobile (PEM) consiste à faire l'évaluation d'une opportunité de mise en marché de billets de spectacles en utilisant certaines fonctionnalités de la plate-forme de négociation générique (GNP2) de l'initiative "Towards Electronic Marketplaces" (TEM) du CIRANO. La particularité de ce projet est d'offrir la réservation des places en envoyant un prix d'offre à partir du téléphone sans fil des clients participants.

Il est divisé en trois volets principaux : Le volet du plan d'affaires, le volet du processus de gestion de l'information et le volet technologique MobiTicket. Cependant nous concentrerons sur le volet technologique dans ce *Work Package*.

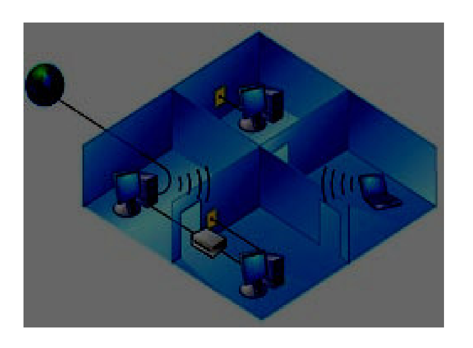
Le Volet Technologique est constitué d'un *Work Package* divisé en cinq sujets selon spécifié ci-dessous :

- Les scénarios de connexion et les besoins de technologie de l'information
- Le cahier de charges
- La politique de modalités des services mobiles
- Les règles des enchères des services mobiles
- Les aspects juridiques des services mobiles

Il est important de souligner que la bibliographie et les ressources Web utilisés pour l'élaboration de tous les documents sont placées à la fin du *Work Package.*

Note : L'emploi du masculin dans les documents présentés sur ce *Work Package*, est utilisé sans discrimination et dans le seul but d'alléger le texte.

2. Les scénarios de connexion et les besoins de technologie de l'information



Le volet technologique MobiTicket

Scénarios de Connexion
Besoins de Technologie de L'information

1. Introduction

Le document qui suit présente quatre scénarios de connexion de l'application PEM comme suit : Connexion Directe, Alternatives à la connexion directe, Plate-forme Cirano et Pan-canadien.

NOTE : Nous nous servirons de textes en français et en anglais dans les sessions 4.1.2 et 4.1.3 qui décrivent les aspects matériel et logiciel des scénarios afin d'économiser du temps de traduction anglais-français.

2. Scénarios de Connexion Directe

Pour la connexion directe nous avons la possibilité d'implémenter deux scénarios tels que connexion via carte AirCard® 555 de Sierra Wireless ou connexion via câble USB.

2.1. Carte Aircard® 555 de Sierra Wireless

Ce scénario nous permet d'envoyer et de recevoir des messages SMS (Short Message Services) directement d'un ordinateur portatif ou de poche au téléphone mobile à travers une Carte sans fil AirCard® 555, jumelée au logiciel "Watcher" et à un forfait Données 1X de Bell Mobilité.

2.1.1. Composants

Pour que nous pussions utiliser les caractéristiques offries pour ce produit de transmission de données sans fil, la carte AirCard® 555, il faut utiliser une trousse d'accès selon décrit ci-dessous :

- Carte AirCard® 555

La Carte sans fil de réseau de radio d'AirCard® 555 de Sierra Wireless est une carte de communication sans fil bi-bande[1] au format PCMCIA[2] est avant tout destinée au marché Nord-Américain et peut fonctionner comme une carte de réseau sans fil, un modem, et un téléphone mobile. Cette carte nous permet d'accéder à l'Internet, d'envoyer et de recevoir des messages SMS et des courriels, de nous connecter à un réseau corporatif, et de faire des appels téléphoniques sans la nécessité d'un câble de réseau ou line de téléphone. Fonctionne, vraiment, comme une carte réseau étendu sans fil, sur le réseau CDMA[3] 2000 1X (haut débit).



Carte AirCard® 555 de Sierra Wireless
Source: Sierra Wireless

La carte AirCard de Sierra Wireless nous permet d'accéder à nos données en glissant la carte dans l'emplacement pour carte PC de notre ordinateur portatif ou de poche.

Pour les services SMS, la carte AirCard® 555 supporte seulement 2-way SMS et pas le broadcast SMS.

Pour obtenir des spécifications techniques sur la carte AirCard® 555 Sierra Wireless, allez voir (en anglais) :
http://www.sierrawireless.com/ProductsOrdering/spec_555.asp

Pour trouver les périphériques compatibles avec la carte AirCard® 550/555, allez voir (en anglais) :
http://www.sierrawireless.com/supportdownload/supported_pdas.asp

- Logiciel "Watcher"

Le logiciel "Watcher" est l'application qui nous permet de gérer et contrôler la connexion entre la carte AirCard® 555 et le réseau CDMA.

Nous utilisons "Watcher" pour débuter des appels voix et de données, determiner la force de notre signal, voir des statistiques et des messages SMS, établir de préférences concernant les messages SMS, l'availabilité de 1X, et d'autres.

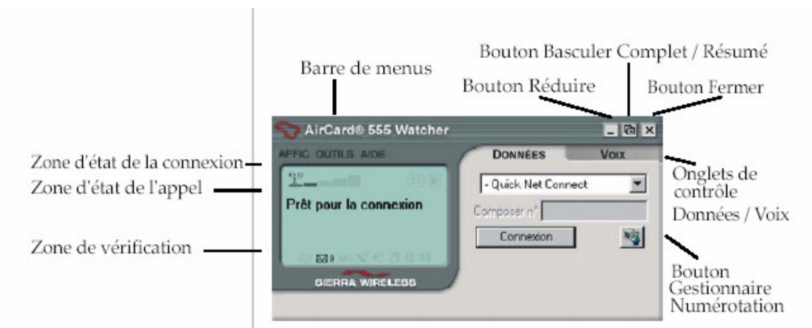
---

[1] Bi-bande : Certains opérateurs transmettent sur la bande de fréquences radio de 900 MHz, tandis que d'autres utilisent la bande 1800 MHz. Les téléphones bi-bande communiquent sur ces deux bandes. Cela élargit votre choix de réseaux à l'étranger, lorsque les accords de l'opérateur couvrent les deux bandes de fréquences.

[2] Carte PCMCIA (Personal Computer Memory Card International Association) : Carte additionnelle, commercialisée dès 1990, conforme aux spécifications PCMCIA (Personal Computer Memory Card International Association). Il s'agit d'un périphérique amovible, de la taille d'une carte de crédit, conçu pour être inséré dans un connecteur PCMCIA. Les cartes PCMCIA peuvent faire office de modem, de fax ou de carte réseau, voire même offrir de la mémoire supplémentaire.

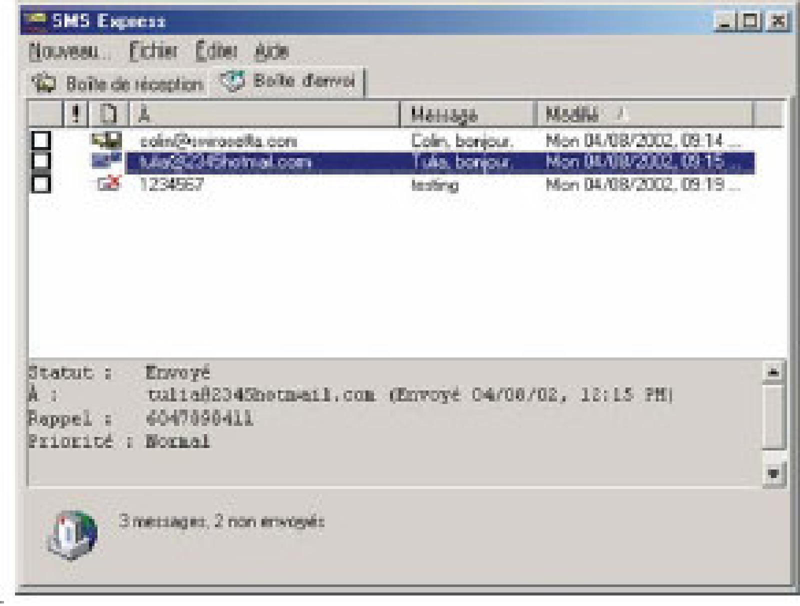
[3] CDMA : Code Division Multiple Access (accès multiple par répartition en code) : Méthode d'accès où les unités mobiles partagent le même canal de radio.

Voici un aperçu des composants de la fenêtre de "Watcher" Window:



Source: Sierra Wireless

Fenêtre SMS Express



Source: Sierra Wireless

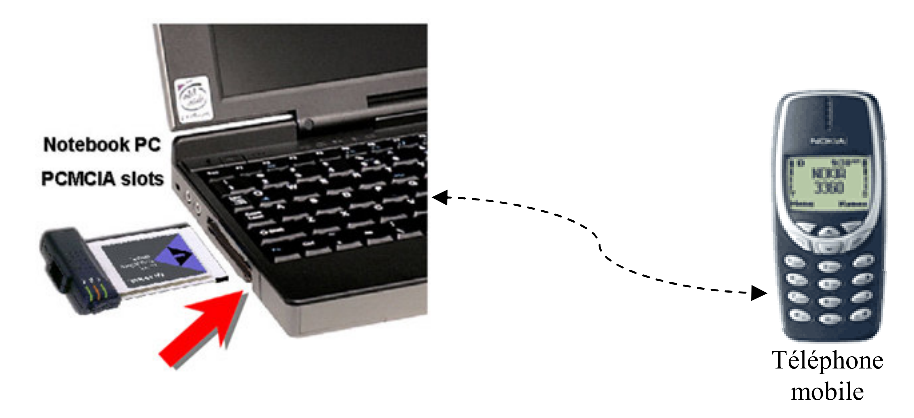
Pour accéder au guide d'utilisateur du logiciel "Watcher", allez voir:
http://www.sierrawireless.com/SupportDownload/docs/2130194FR.pdf

- Forfait Données 1X

Pour l'utilisation des services l'utilisateur doit s'abonner au forfait Données 1X qui est un forfait de transmission de données de Bell Mobilité

La technologie One times (1X) est l'évolution 3G de la technologie AMRC (accès multiple à répartition par code), la norme privilégiée pour les télécommunications sans fil en Amérique du Nord. 1X est une technologie à haute vitesse efficace qui double les capacités de transmission de la voix et la vitesse de transmission des données par paquet et qui prolonge la durée des piles de téléphones.

2.1.2. Schéma de la connexion



La carte AirCard de Sierra Wireless nous permet d’accéder à nos données en glissant la carte dans l’emplacement pour carte PC de notre ordinateur portatif ou de poche.



Sierra Wireless AirCard® 555
for Verizon with Compaq iPAQ
Source: Sierra Wireless

2.2. Câble USB

Composants

Data suite USB chargeur : Câble et CD ROM d'installation pour Motorola V60/V66/V70 : envoi et réception des SMS/fax/email - fonction modem - fonction chargeur pour le mobile, transfert de sonneries et logos, synchronisation de l'agenda (édition du répertoire), et d’autres.

Schéma de la connexion



Branchement sur le câble USB du PC
Source : Motorola Inc.

2.3. IPAQ 3800 CABLE (REF S9908)

Composants

Connexion de la série iPAC H38XX Pocket PC handheld vers un téléphone Motorola compatible. Note : Il faut avoir l'accès Internet ou l'application courriel pour le Pocket PC pour naviguer sur le Web et pour envoyer et recevoir des courriels avec ce produit.

Schéma de la connexion



IPAQ 3800 Cable (Ref S9908). Source : Motorola Inc.

2.4. Connexion du HP Jornada Pocket PC Handheld

Connexion du HP Jornada Pocket PC handheld vers un téléphone Motorola compatible. Accès de données en tout temps et en tout lieu.



HP Jornada Cable (ref S9909). Source : Motorola Inc.

2.5. Nokia 3585 Data Product

Le câble de données DKU-5 nous permet d'utiliser notre téléphone comme un modem sans fil et de synchroniser notre téléphone avec un PC. Ce câble est compatible avec des téléphones CDMA. Note: Des « software driver files » sont requis pour l'installation.



Nokia 3585 Data Product**:** Câble de données DKU-5
Source : Nokia Inc.

2.6. NOKIA 6185i



Téléphone Nokia 6185i : Câble de données (DLR-3P)
Source : Nokia Inc.

3. Scénarios de Solutions Alternatives à la Connexion Directe

3.1. Environnement Intégré de Développement Zim

La compagnie Zim est un agrégateur qui est branché à Bell et qui offre une trousse de connectivité, et un API UNIX. ZimWeb est une solution de connectivité qui assure que de données peut être accédé, mise à jour, et maintenue via l'Internet en utilisant une variété de clients qui sont équipés avec un navigateur (c.f. PCs, PDAs, téléphone cellulaire,etc.) Pour l'utilisation de cette solution il faudra demander un accès via Bell, si la compagnie ZIM est partenaire de Bell.

Pour plus d'informations concernant les spécifications techniques de ZimWeb, allez voir: http://www.zim.biz/collateral/ZL_FS_ZimWeb_01.03.pdf

3.2. SMSXCHANGE

Le SMSXchange.com c'est un portail qui fournit des services d'impartition SMS à travers les options suivantes : Interface d'application HTTP, Interface des courriels, ou Hébergement dédié.

Pour ce service encourent des frais d'installation (USD$2000) et aussi des frais mensuels (USD$150) pour chaque serveur. Pour plus d'informations, allez voir: http://www.smsxchange.com/main/sms_delivery.asp

Pour plus d'informations concernant les spécifications techniques, allez voir:
http://www.smsxchange.com/account/whitepaper.pdfhttp://www.zim.biz/collateral/ZL_FS_ZimWeb_01.03.pdf

3.3. SMS SERVER 2.0 (ÉVALUATION)

C'est un logiciel SMS GATEWAY basé sur des modems GSM disponible en évaluation sans limite d'expiration. Il envoie des messages SMS à travers de téléphones mobiles Nokia.
Le SMSServer 1.0 s'execute seulement sur les suivants telephones Nokia:

- Nokia 3210
- Nokia 3310
- Nokia 3330
- Nokia 81xx
- Nokia 8210
- Nokia 51xx
- Nokia 61xx
- Nokia 6210
- Nokia 7110

Note : Il faut aussi avoir le câble sans fil correspondent aux modèles indiqués ci-dessus.

3.4. Service Courriel de Bell Mobilité

Il faut s'abonner au service. Pour plus d'informations, allez voir:
http://www.bell.ca/shop/application/commercewf?origin=noorigin.jsp&event=link(goto)&content=/jsp/content/personal/catalog/wireless/data/wireless_email/applications/relay.jsp&contentpage=relay

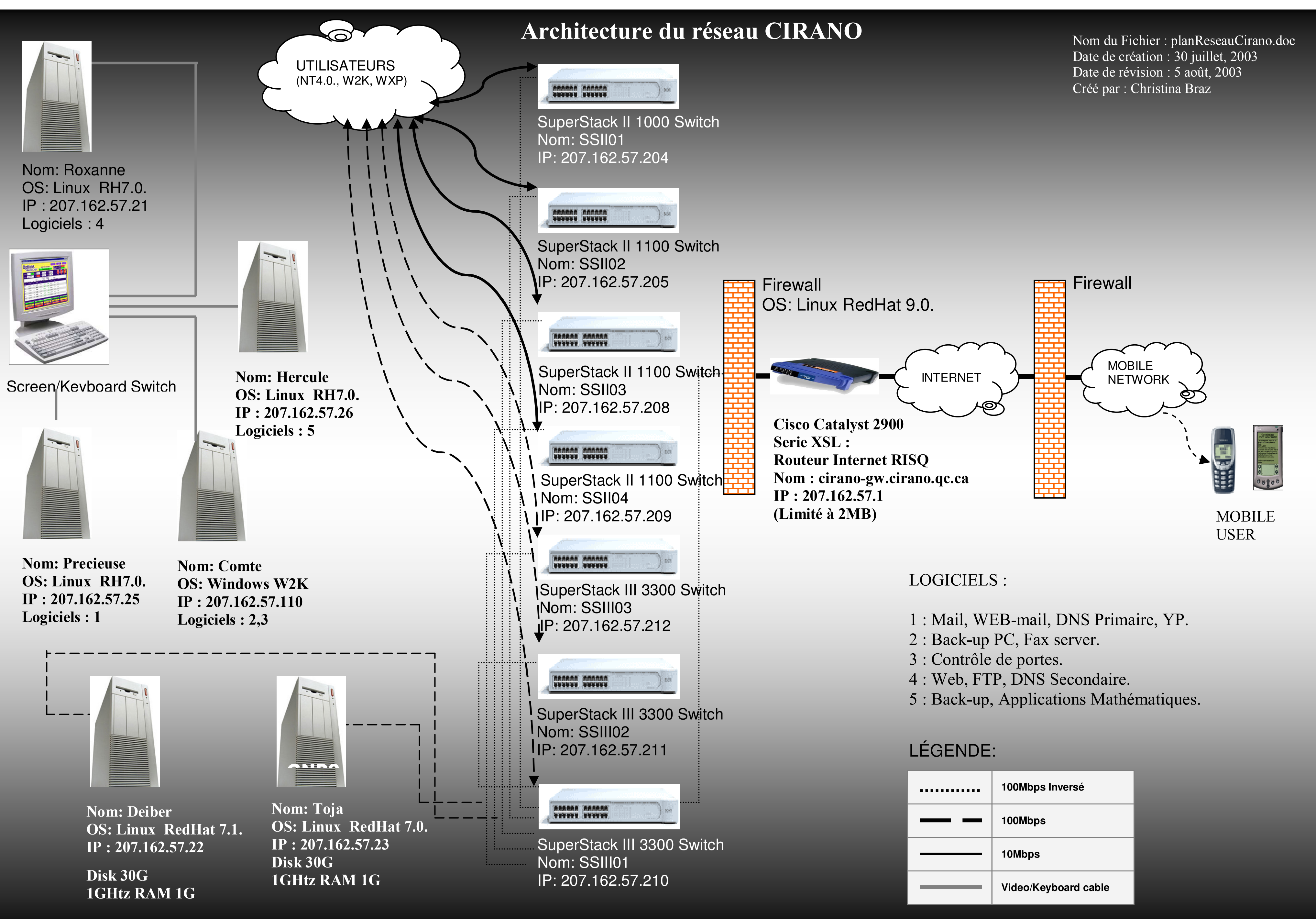
4. Scénario Réseau CIRANO

Pour la connexion via Réseau Cirano nous décrirons l'architecture du réseau et le matériel nécessaire pour le déploiement d'envoi d'à partir 200,000 messages SMS au Québec.

Note : Considérant que nous avons environ 10 millions d'abonnés mobiles au Canada, que les usagers potentiels du PEM représentent 4% de ce total, soit 400,000 abonnés mobiles, et qu'au Québec ce total atteints 200,000 abonnés.

4.1.1. Architecture du Réseau CIRANO

Voir la prochaine feuille.



Architecture du réseau CIRANO
Nom du Fichier : planReseauCirano.doc
Date de création : 30 juillet, 2003
Date de révision : 5 août, 2003
Créé par : Christina Braz
UTILISATEURS
(NT4.0., W2K, WXP)
Nom: Roxanne
OS: Linux RH7.0.
IP : 207.162.57.21
Logiciels : 4
Screen/Keyboard Switch
Nom: Hercule
OS: Linux RH7.0.
IP : 207.162.57.26
Logiciels : 5
Nom: Precieuse
OS: Linux RH7.0.
IP : 207.162.57.25
Logiciels : 1
Nom: Comte
OS: Windows W2K
IP : 207.162.57.110
Logiciels : 2,3
Nom: Deiber
OS: Linux RedHat 7.1.
IP : 207.162.57.22
Disk 30G
1GHtz RAM 1G
Nom: Toja
OS: Linux RedHat 7.0.
IP : 207.162.57.23
Disk 30G
1GHtz RAM 1G
SuperStack II 1000 Switch
Nom: SSII01
IP: 207.162.57.204
SuperStack II 1100 Switch
Nom: SSII02
IP: 207.162.57.205
SuperStack II 1100 Switch
Nom: SSII03
IP: 207.162.57.208
SuperStack II 1100 Switch
Nom: SSII04
IP: 207.162.57.209
SuperStack III 3300 Switch
Nom: SSIII03
IP: 207.162.57.212
SuperStack III 3300 Switch
Nom: SSIII02
IP: 207.162.57.211
SuperStack III 3300 Switch
Nom: SSIII01
IP: 207.162.57.210
Firewall
OS: Linux RedHat 9.0.
Cisco Catalyst 2900
Serie XSL :
Routeur Internet RISQ
Nom : cirano-gw.cirano.qc.ca
IP : 207.162.57.1
(Limité à 2MB)
INTERNET
Firewall
MOBILE NETWORK
MOBILE USER
LOGICIELS :
1 : Mail, WEB-mail, DNS Primaire, YP.
2 : Back-up PC, Fax server.
3 : Contrôle de portes.
4 : Web, FTP, DNS Secondaire.
5 : Back-up, Applications Mathématiques.
LÉGENDE:
100Mbps Inversé
100Mbps
10Mbps
Video/Keyboard cable

4.1.2. Caractéristiques Techniques des Matériels Utilisés

Pour une meilleure compréhension et une complétude des informations techniques qui seront décrites dans ce document, nous nous utiliserons de textes en français et en anglais puisque l'offre de ressources en anglais est plus vaste qu'en français.

Description	*Function*	*Name*	*Qty*	*Characteristics*
Cisco Catalyst	Router Internet : A device on a network that examines data coming in and determines where to send it on to its ultimate destination.	RISQ	1	The Cisco Catalyst® Series of switches consists of a comprehensive range of Ethernet-based platforms providing increasing levels of functionality from basic connectivity to high-end, multilayer intelligent service switching solutions. These switches provide diverse interfaces, high port densities and extensible functionality suitable for converged network access or backbone applications. As a key component of Cisco AVVID (Architecture for Voice Video and Integrated Data), Catalyst switches provide the scalable network infrastructure with intelligent services that enables organizations to deploy converged Internet business solutions to maximize their productivity and competitive advantage. The Catalyst® 2900 family has the industry's highest density, small form factor fixed configuration switches, offering feature-rich end-to-end software and solutions for workgroup and wiring closets.
Firewall (Clone d'IBM)	Control outgoing and incoming network traffic based on corporate security policy.	n/a [1]	1	Operating System: Linux Red Hat 9.0.
3COM Super Stack II 1000 Switch	Switch: A device that facilitates transmissions between nodes on a private network and regulates internal network traffic.	n/a	1	The 3Com SuperStack II Desktop Switch is a 24-port 10 Mbps Ethernet Switch, with one 100BASE-TX Fast Ethernet port, a plug-in module slot for an additional high-speed interface (Fast Ethernet, ATM OC-3), and a transceiver module slot for a 10 Mbps Ethernet uplink. The SuperStack switches are the only units we found in the workgroup switch category that support ATM. This 19" rack mountable unit provides dedicated 10 Mbps connections for individual end stations ("power users" and servers). Both switches are also stackable. The Desktop Switch supports Virtual LAN configurations, four packet forwarding modes, flow management, and the SNMP agent in the switch supports industry standard Remote Monitoring (RMON) MIB modules.The aptly named Desktop Switch supports one end station per port on all ports other than those designated as backbone ports, and can optionally support a separate spanning tree protocol (STP) for bridging per VLAN.
3COM Super Stack II 1100 Switch	Switch: A device that facilitates transmissions between nodes on a private network and regulates internal network traffic.	n/a	3	The SuperStack II Switch 1100 is perfect for desktop connectivity and its ability to scale to larger configurations makes it equally suited for handling hub aggregation. A single device can support up to 6,000 MAC addresses. The SuperStack II Switch 1100 is available in 12- and 24-port versions and features two built-in auto-sensing 10/100 Fast Ethernet ports. All models

				have a matrix port on the rear of the unit that enables high-speed connection to other SuperStack II Switch 1100 and Switch 3300 units. This eliminates the bottlenecks caused by interconnecting switches with Fast Ethernet ports. The stacked switches can be managed as a single entity and share a single IP address. As with other SuperStack II switches, the new SuperStack II Switch 1100 feature auto-sensing 10/100 ports that adjust for 10BASE-T and 100BASE-TX attached devices. Switch 1100 automatically provides full-duplex/half-duplex capability on all ports to boost bandwidth for servers and power users. Plus, they both help you implement advanced policy-based management across the network with features such as support for Fast IP, IGMP snooping, IEEE 802.1p prioritization, and IEEE 802.1Q standards-based VLANs. The SuperStack II Switch 1100 and Switch 3300 can be stacked up to four units high by using the SuperStack II Switch Matrix Module and SuperStack II Switch Matrix Cables: Mix and match Switch 1100 and Switch 3300 within the stack to meet customer needs. Stack up to four units high supporting up to 104 switched ports. SuperStack II Switch Matrix Module supports 4 x 1 Gbps links between switches to create high-density switches without wasting Fast Ethernet or Gigabit Ethernet ports.
Description	*Function*	*Name*	*Qty*	*Characteristics*
3COM Super Stack III 3300 Switch	Switch: A device that facilitates transmissions between nodes on a private network and regulates internal network traffic. Auto-sensing Switching for Workgroups and Backbones	n/a	3	Total ports: 24 auto-sensing 10/100 Ethernet, 1 matrix, 1 expansion slot Media Interfaces: 10/100BASE-TX/RJ-45; optional 100BASE-FX/SC, 1000BASE-FX 1000BASE-T/RJ-45, 1000BASE-SX/SC, 100BASE-LX/SC. Switching Features: Store-and-forward, full-/half-duplex auto-negotiation, port trunking, 802.1Q VLAN support, 802.1p traffic prioritisation. Managment: Embedded Web server for direct management Dimensions: (HxWxD) 7 cm x 44 cm x 30 cm
Unix Server (Clone d'IBM)	An application that answers and fulfills client request. Examples of servers include a DBMS server that provides connectivity to a relational database and an HTTP server that provides HTML pages and services HTTP requests.	Precieuse	1	A 1-way or 2-way symmetric multiprocessor (SMP) server with state-of-the-art, 64-bit, copper-based POWER4+ microprocessors, providing enterprise availability for mission critical requirements and outstanding performance for commercial and technical computing requirements. Processor Intel AMD Pentium III processor Clock rates (Min/Max) 1.2GHz System memory (Std/Max) 1GB / 16GB

				Internal storage (Std/Max) 36.4GB / 1.1TB Performance (rPerf range) 2.50 to 4.00h
Unix Server (Clone d'IBM)	Same above.	Comte	1	Same above.
Unix Server (Clone d'IBM)	Same above.	Hercule	1	Same above.
Unix Server (Clone d'IBM)	Same above.	Roxanne	1	Same above.
Unix Server (Clone d'IBM)	Same above.	Deiber	1	Same above.
Unix Server (Clone d'IBM)	Same above.	Toja	1	Same above.
IBM Monitor/Keyboard/ Mouse Switch	This server selector switch allows the user, with a special key sequence, to select which server is connected to the console devices. An on-screen facility indicates to the user which server is currently using the monitor, keyboard, and pointing device.	n/a	1	Connects up to eight PC Servers to a single console. Sits between the video monitor, keyboard, and pointing devices and the server system units. Multiple switch units can be interconnected in a two-level tier arrangement to support up to 64 servers. A primary switch can support up to eight secondary switches which each support up to eight server systems. The operator controls the switch through the use of the console keyboard and the monitor. The server selector operates independently of the server's operating system; therefore, different operation systems may be used on the different server in the same rack suite. The server selector switch manages the console communications with all of the servers independently and simultaneously. It can be optionally housed behind the monitor compartment or in the special compartment in the rack side wall to save vertical rack mounting space. With this product you enjoy a one year Customer Carry-in Exchange.

[1] Not applicable.

4.1.3. Caractéristiques Techniques des Logiciels Utilisés

Pour une meilleure compréhension et une complétude des informations techniques qui seront décrites dans ce document, nous nous utiliserons de textes en français et en anglais puisque l'offre de ressources en anglais est plus vaste qu'en français.

La majorité des logiciels utilisés au Cirano sont gratuits (logiciels libres), sauf s'ils sont indiqués avec un astérisque selon le tableau ci-dessous :

Description	*Fonction*	*Marque (Vendor)*	*Caractéristiques*
Mail	Le programme utilisé dans le monde Internet pour la transmission du courrier électronique.	SendMail	Sendmail est un MTA (agent de transfert de messages), il envoie des messages, reçoit, route (relayage), fait délivrer les, messages dans les boîtes aux lettres, gère la file d'attente. Il est souple et éminemment configurable, conçu pour acheminer de façon fiable les messages d'un réseau à un autre, au point qu'il s'agit du standard de fait constituant le cœur de la plupart des mises en œuvre du protocole SMTP et des autres protocoles liés à la transmission de messages sur Internet.
WEB-mail	Service de messagerie électronique complet, utilisable directement en ligne de n'importe quel ordinateur relié à Internet. Il suffit de se connecter à l'adresse du service puis de rentrer son identifiant et son mot de passe. C'est un service très pratique lorsqu'on est amené à se déplacer, pour un usage nomade, personnel ou professionnel.	Twig (The Web Information Gateway)	TWIG is a groupware client designed to be used with web browsers. The main goal was to provide an IMAP client, but it now offers agenda, news-reader,bookmarks, contact list and other features. TWIG can be used as a webmail app with remote mail and news servers, so these are not needed in the local machine. This package is prepared to be used with either MySQL or PostgreSQL, but it is possible to use it with many other databases if properly configured.
DNS Primaire/DNS Secondaire	DNS (Système de noms de domaine) : la norme Internet de conversion de noms d'hôte en adresses numériques IP. Un serveur de noms de domaine est un ordinateur (serveur) qui enregistre l'adresse Internet correspondant à toute adresse URL que nous tapons ou sur laquelle nous cliquons à partir de notre navigateur. Il y a habituellement deux adresses de serveur (primaire et secondaire).	BIND (Berkeley Internet Name Daemon) Version 9	DNS Security • DNSSEC (signed zones) • TSIG (signed DNS requests) IP version 6 • Answers DNS queries on IPv6 sockets • IPv6 resource records (A6, DNAME, etc.) • Bitstring Labels • Experimental IPv6 Resolver Library DNS Protocol Enhancements • IXFR, DDNS, Notify, EDNS0 • Improved standards conformance Views • One server process can provide multiple "views" of the DNS namespace, e.g. an "inside" view to certain clients, and an "outside" view to others. Multiprocessor Support Improved Portability Architecture.

			BIND RESOURCE REQUIREMENTS *Hardware requirements* BIND 9 currently requires a UNIX system with an ANSI C compiler, basic POSIX support, and a 64 bit integer type. DNS hardware requirements have traditionally been quite modest. For many installations, servers that have been pensioned off from active duty have performed admirably as DNS servers. The DNSSEC and IPv6 features of BIND 9 may prove to be quite CPU intensive however, so organizations that make heavy use of these features may wish to consider larger systems for these applications. BIND 9 is now fully multithreaded, allowing full utilization of multiprocessor systems for installations that need it. *CPU Requirements* CPU requirements for BIND 9 range from i486-class machines for serving of static zones without caching, to enterprise-class machines if you intend to process many dynamic updates and DNSSEC signed zones, serving many thousands of queries per second. *Memory Requirements* The memory of the server has to be large enough to fit the cache and zones loaded off disk. *Supported Operating Systems* ISC BIND 9 compiles and runs on the following operating systems: • IBM AIX 4.3 • Compaq Digital/Tru64 UNIX 4.0D • Compaq Digital/Tru64 UNIX 5 (with IPv6 EAK) • HP HP-UX 11 • IRIX64 6.5 • Sun Solaris 2.6, 7, 8 • NetBSD 1.5 (with unproven-pthreads 0.17) • FreeBSD 3.4-STABLE, 3.5, 4.0, 4.1 • Red Hat Linux 6.0, 6.1, 6.2, 7.0
Partage de comptes utilisateurs en environnement Linux.	NIS/YP (Network Information Services/Yellow Pages) est un système client/serveur basé sur les RPCs qui permet à un groupe de machine d'un domaine NIS (Unix) de partager un ensemble de fichiers de configuration communs. L'administrateur système peut ainsi mettre en place des clients NIS en n'ayant qu'un minimum de configuration à faire et ajouter, modifier ou supprimer des informations	NIS/YP	PACKET FILTERING CHARACTERISTICS OF NIS/YP NIS/YP is an RPC-based service, generally provided over UDP. It's very difficult to handle RPC-based services with a packet filtering system, because the servers normally don't use predictable port numbers. While the port numbers to be used are too unpredictable for a packet filtering system to cope with, they're not so unpredictable that an attacker can't find them. (If nothing else, the attacker could try sending NIS/YP requests to all ports, to see which respond as they would expect an NIS/YP server to.) PROXYING CHARACTERISTICS OF NIS/YP RPC-based protocols are almost as unpleasant to proxy as they are to allow with packet filtering; they cannot be adequately handled with generic proxies. A dedicated NIS/YP proxy server

	de configuration de façon centralisée.		would be possible, but we do not know of one, except for the proxying capabilities available with NIS+. In any case, proxying alone does not adequately deal with the vulnerabilities of NIS/YP. PHYSICAL SERVER REQUIREMENTS There are several things to keep in mind when choosing a machine to use as a NIS server. One of the unfortunate things about NIS is the level of dependency the clients have on the server. If a client cannot contact the server for its NIS domain, very often the machine becomes unusable. The lack of user and group information causes most systems to temporarily freeze up. With this in mind you should make sure to choose a machine that will not be prone to being rebooted regularly or one that might be used for development. The NIS server should ideally be a stand alone machine whose sole purpose in life is to be an NIS server.
Back-up PC	A copy of a file, directory, or volume on a separate storage device from the original, for the purpose of retrieval in case the original is accidentally erased, damaged, or destroyed.	Unix (Windows 2000)	• Handles files which lengthen or shorten during backup • Exclude all read-only mounted filesystems optionally • Can limit backup to a single filesystem without unmounting • Verification of full backup to ensure restorability • Pre-configured profiles for ease of use • Full security embedded into the backup • On-line trouble-shooter which generates its own technical support fax sheet • Backup results and verification mailed to the system administrator • Ability to set the level of subdirectories to backup • Double buffering (splits backup into 2 processes which helps improve backup time to 35-50%, especially fast for multiple CPU environments) • Speed option using shared memory, semaphores, and split processes
Fax Server	WinFax is a fax management software for businesses. • Organizes and manages faxes. • Sends and receives professional-looking faxes. • Integrates with key business applications.	WinFax Pro 10.0 [*]	Platforms: Windows® (XP Home/XP Pro/2000 Pro/Me/98). Windows NT/2000 Server, Advanced Server and Terminal Servers are not supported.
Contrôle de portes	EntraPass is a multi-user access control system allowing users to simultaneously access the system from up to 128 workstations connected to the network.	EntraPass [*]	• Network ready • Up to 4,096 card readers • SmartLink integration • Up to 100 alarm-system partitions per site • Dynamic color • Visual controller diagnostics • Compatible Windows
Serveur Web	Système fournissant un service à un ou des clients potentiels; un serveur http (Web) fournit des pages HTML et les fichiers référencés dans ces pages (images, applets,etc.). Un	Apache Jakarta Project	Apache: projet libre de développement logiciel dont l'objectif est de créer un serveur HTTP (Web) robuste, riche en fonctionnalités et au code source librement diffusable. Serveur http le plus répandu, issu du 'logiciel libre'. Est complété sur un site Web par des logiciels fabriquant des pages dynamiques, en utilisant des technologies variées: CGI(vieux),

	serveur de base de données fournit les informations contenues dans sa ou ses bases.		PHP, JSP, XML.
FTP (File Transfer Protocol)	Le logiciel WS_FTP implante sur notre micro-ordinateur le protocole FTP de TCP/IP. Comme son nom le laisse entendre, ce protocole nous permet de nous connecter à un autre ordinateur en utilisant un numéro de compte pour télécharger (de notre micro-ordinateur à un autre ordinateur) et pour télédécharger (d'un autre ordinateur à notre micro-ordinateur) des fichiers.	WS_FTP Ipswitch	• Environnement : Windows NT/95/98 • Version 7.0 • Taille : 3,88 Mo
Dump (File system Back-up)	A backup or a backup file created by the UNIX dump utility. The dump utility examines files on a file system and determines which files need to be backed up. These files are copied to the given disk, tape or other storage medium for safe keeping.	Dump (Unix)	• A dump utility appeared in Version 6 AT&T UNIX. • This usage is typical only at large time-sharing installations.
Applications Mathématiques	Logiciel de calcul scientifique et technique.	Mathematica [*], MatLab [*], Gauss[*]	• Mathematica : Voir: http://www.logilabo.com/prods/mathematica.html • MatLab: Voir: http://www-curri.u-strasbg.fr/DOCtechnique/matlab.htm • Gauss: Voir : http://gurukul.ucc.american.edu/econ/gaussres/GAUSSIDX.HTM
Serveur d'application	Serveur hébergeant les applications destinées à être utilisées dans un réseau distribué.	BEA WebLogic Server 6.1	Web and Wireless application server, powers the most sophisticated e-business applications. WebLogic Server's advanced architecture implements the Java 2 Enterprise Edition (J2EE™) platform specification, including Servlets, Java Server Pages (JSP), Enterprise JavaBeans (EJB™), Java Messaging Services (JMS), and other platform services that provide the scalability, flexibility, and reliability required by multi-tier, mission-critical e-business applications. • WebLogic Server Compatibility • WebLogic Tuxedo Connector • WebLogic Messaging Bridge • J2EE Connector Architecture • Java 2 Security • JTA 1.0.1. • EJB 2.0 (including support for message-driven beans) • JMS 1.0.2 • Servlet 2.3 • JSP 1.2 • In-Memory Replication of Stateful Session EJBs • JMX-based System Management

			• Deployment Descriptor Editor
Outil de firewalling	Firewalling : reglès de filtrage de paquets. L'application IpTables est une solution complète de firewall (noyau 2.4) tournant sous le système GNU/Linux.	IPTables (Linux)	Depuis la version 2.4, Linux contient un module destiné au filtrage réseau, **Netfilter**. Il se configure au moyen d'un outil appelé IPTables. Le filtrage réseau consiste en l'examen des paquets réseaux et à prendre des décisions sur le traitement à leur appliquer. C'est ce que fait un **pare-feu**. Avec un système GNU/Linux, pour configurer des règles de pare-feu, il faudra donc simplement utiliser Netfilter à l'aide d'IPTables. IpTables permet de faire du firewalling *stateful* (à états), de la translation de port et d'adresse, de modifier des règles de filtrage, etc. Le filtrage de paquets IP est sans doute la base du *firewalling*.
Gestionnaire de Servlets	Tomcat est un logiciel "open-source", et généralement utilisé en couplage avec un un serveur Web tel que Apache, et qui permet d'exécuter des programmes serveur et des pages Web dynamiques (servlets Java).	Tomcat	Tomcat répond aux spécifications de Sun concernant les servlets et leurs gestionnaires. Ecrit en Java, il nécessite, pour pouvoir fonctionner, la présence d'une machine virtuelle Java, et plus précisément du SDK - Sun Developpement Kit - complet. Ceci entraîne le fait que Tomcat est totalement portable et peut être mis en oeuvre sur des systèmes radicalement différents, tels que Linux ou Windows.
Système d'exploitation Linux	Le système d´exploitation (SE) dirige tous les autres programmes, appelées les applications, qui font des demandes de services par une interface d'un programme d'application défini (IPA). Un utilisateur peut dialoguer directement avec le SE par une interface (par exemple, un ordre de gestion). Linux (GNU/Linux) est un SE de type Unix développé sous forme de logiciel libre.	Linux	• Stable : Linux offre une base solide car assure en continu un bon service : pas « de plantage surprise » du système ni d'instabilité de l'ensemble dûe aux errements de certains programmes. • Nombreux logiciels interopérables, souples et extensibles : De nombreux logiciels libres coopèrent facilement, sont facilement adaptables à de nouveaux besoins, et offrent un langage de programmation grâce auquel l'automatisation des traitements et l'ajout de nouvelles fonctions respecte au mieux les exigences de l'utilisateur tout en demeurant relativement simple. • Code source disponible Cela facilite l'apprentissage et la maintenance corrective (correction de bugs) ou adaptative (extension des fonctions assurées). • Multi-utilisateur (*multi-user)* et multitâche *(multi-task)*: Il est un système pour plusieurs utilisateurs et processus (orienté vers le réseau). C'est-à-dire que le système accepte que plusieurs utilisateurs travaillent en même temps sur une station (via des terminaux textes ou graphiques). En plus chaque utilisateur peut effectuer plusieurs tâches en même temps. • Temps partagé *(time sharing)*: c'est à dire que les ressources du processeur et du système sont réparties entre les utilisateurs. • Entrées/Sorties intégrées au système de fichiers: les périphériques sont représentés par des fichiers - cela rend le système indépendant du matériel et en assure la portabilité - l'accès aux périphériques est donc identique à l'accès aux fichiers ordinaires. • X Window : X Window est un système de fenêtrage (pour interface

			graphique) souple, puissant et robuste. Il permet, grâce au réseau, d'utiliser sur une machine une application graphique fonctionnant sur une autre. N'importe quel poste de travail connecté au réseau dispose ainsi des ressources de tous les autres. • Performances et rendement : Linux, à configuration matérielle égale, s'avère très souvent plus rapide que les autres systèmes d'exploitation. Une seule machine peut satisfaire les besoins de plusieurs utilisateurs simultanément connectés. Il fonctionne de façon utile même sur de modestes configurations, grâce à sa modularité, et fonctionne donc sur des machines obsolètes.
Red Hat Linux Version 7.1.	Red Hat Linux 7 contient tout ce qui est nécessaire à l'installation d'un poste de travail ou d'un serveur.	Red Hat	• Langages de programmation multiples (C, C++, FORTRAN, Perl, Python, TCL et GCJ compilateur GNU pour Java). • Série d'outils de développement mise àjour avec GCC 2.96, GDB 5.0, GLIBC 2.2 • Outils de développements d'applications Web tels que PHP et Zope • OpenSSL avec cryptage 128 bits pour une sécurité des communications Web, • Support USB pour les souris et claviers • XFree 4.0.1 pour une meilleure performance vidéo • Outil de configuration de pare-feu graphique • Service de mise à jour automatique de systèmes via le Net, etc.
Red Hat Linux Version 9.0.	Red Hat Linux 9.0. contient tout ce qui est nécessaire à l'installation d'un poste de travail ou d'un serveur.	Red Hat	• Red Hat Linux 9 fournit les dernières technologies Linux de la communauté Open Source dans un système d'exploitation facile à utiliser. • Red Hat Linux comprend un système d'installation plus puissant, supporte des options plus complètes et contient le support pour du clustering avancé. • Applications incluses : Ximian EvolutionTM pour le courrier électronique, le calendrier et le carnet d'adresse. • Utilise les dernières technologies Red Hat Linux avec l'interface graphique Bluecurve. HARDWARE REQUIREMENTS The following information represents the minimum hardware requirements necessary to successfully install Red Hat Linux 9: • CPU: - Minimum: Pentium-class - Recommended for text-mode: 200 MHz Pentium-class or better - Recommended for graphical: 400 MHz Pentium II or better • Hard Disk Space: - Custom Installation (minimum): 475MB - Server (minimum): 850MB - Personal Desktop: 1.7GB - Workstation: 2.1GB - Custom Installation (everything): 5.0GB • Memory: - Minimum for text-mode: 64MB - Minimum for graphical: 128MB - Recommended for graphical: 192MB

Struts 1.0.2	Struts est un Framework pour faire une interface humain-machine avec un modèle MVC pour développer des applications Web. Framework : architecture et ensemble d'outils de développement. Struts est un projet Open Source développé dans le cadre du projet Jakarta de l'ASF (Apache Software Foundation).	Apache Jakarta Project	How Struts works? Java Servlets are designed to handle requests made by Web browsers. Java ServerPages are designed to create dynamic Web pages that can turn billboard sites into live applications. Struts uses a special Servlet as a switchboard to route requests from Web browsers to the appropriate ServerPage. This makes Web applications much easier to design, create, and maintain.
Servlet 2.3	Programme écrit en Java qui s'exécute sur un serveur, par opposition à une applet qui s'exécute dans le navigateur. Classes Java qui génèrent du contenu dynamique.	Sun Microsystems	There have been many changes in the specification between the 2.2 final version and the 2.3 proposed final draft. Some of the changes include : • incorporating Javadoc API definitions into the specification, • making the J2SE a requirement for the underlying platform for web containers, • internationalization fixes, and • incorporation of many of the Servlet 2.2 errata and other clarifications.
JSP1.2	Java Server Pages pour les pages dynamiques.	Sun Microsystems	• JSP 1.2 is based on Servlet 2.3 and Java 2. • The include action can now be used without flushing. • The XML syntax for a JSP page has been finalized. • Tag libraries can make use of Servlet 2.3 event listeners. • A new type of page validator has been added. • New options for tag library distribution and deployment have been added. • Two new tag interfaces have been added. • The conversion rules for String literal tag attribute values now include conversion to the Object type. • A PropertyEditor can be used for custom attribute value conversion. • The tag handler lifecycle has been clarified. • The Tag Library Descriptor has been aligned with other J2EE descriptors and extended with new elements.
JMS 1.0.2	JMS, acronyme de Java Messaging Service, est une API fournie par Sun pour permettre un dialogue standard entre des applications ou des composants via des brokers de messages ou MOM (Middleware Oriented Messages).	Sun Microsystems	Permet l'envoi asynchrone de messages et un séquencement du traitement de ceux-ci. Elle permet donc d'utiliser des services de messaging dans des applications java comme le fait l'API JDBC pour les bases de données. Des informations utiles sur JMS peuvent être trouvées à l'URL : http://java.sun.com/products/jms/index.htm
EJB 2.0	Permet la persistance dans la base de données (entity beans), la réception (message-driven beans) et l'envoi (session beans) des requêtes. Les Entreprise Java Bean ou EJB sont des composants	Sun Microsystems	Une des principales caractéristiques des EJB est de permettre aux développeurs de se concentrer sur les traitements orientés métiers car les EJB et l'environnement dans lequel ils s'exécutent prennent en charge un certains nombres de traitements tel que la gestion des transactions, la persistance des données, la sécurité, etc. Les EJB s'exécutent dans un environnement particulier : le serveur d'EJB. Celui ci fournit un ensemble de fonctionnalités

	serveurs donc non visuel qui respectent les spécifications d'un modèle édité par Sun.		utilisées par des conteneurs d'EJB qui constituent le serveur d'EJB. Un serveur peut contenir plusieurs conteneurs. En réalité, c'est dans un conteneur que s'exécute une EJB.
Castor 0.9.3.9	Sérialise les objets économiques en format XML ce qui permet de les stocker dans la base de données.	Apache Jakarta Project	It is basically the shortest path between Java objects, XML documents and SQL tables. Castor provides Java to XML binding, Java to SQL persistence, and then some more.
Xdoclet 1.2	Permet la génération d'interfaces et de descripteurs de déploiement pour les EJB. XDoclet is a code generation engine. It enables *Attribute-Oriented Programming* for Java. In short, this means that we can add more significance to our code by adding meta data (attributes) to our Java sources. This is done in special JavaDoc tags.	Apache Jakarta Project	XDoclet will parse our source files and generate many artifacts such as XML descriptors and/or source code from it. These files are generated from templates that use the information provided in the source code and its JavaDoc tags. XDoclet lets us apply Continuous Integration in component-oriented development. Developers should concentrate their editing work on only one Java source file per component. Currently XDoclet can only be used as part of the build process utilizing Jakarta Ant .XDoclet comes with a set of modules for generation of different kinds of files. Users and contributors can write their own modules (or modify existing ones) if they wish to extend the functionality of XDoclet.
JBoss 3.0.6	Serveur d'application (alternative à WebLogic). JBoss est un projet open source développé en Java pour fournir un environnement d'exécution d'EJB respectant les spécification J2EE.	JBoss	JBoss est composé d'un ensemble d'outils : JBoss Server, JBoss MQ (implémentation de JMS), JBoss MX, JBoss TX(implémentation de JTA/JTS), JBoss SX , JBoss CX et JBoss CMP. JBoss provides JBossServer, the basic EJB container and JMX infrastructure. JBossMQ, for JMS messaging, JBossMX, for mail, JBossTX, for JTA/JTS transactions, JBossSX for JAAS based security, JBossCX for JCA connectivity and JBossCMP for CMP persistence. JBoss enables you to mix and match these components through JMX by replacing ANY component you wish with a JMX compliant implementation for the same APIs. JBoss doesn't even impose the JBoss components, that is modularity.
Log4J 1.2	Gestion des logs produits par les différentes entités de la plateforme GNP2.	Apache Jakarta Project	With log4j it is possible to enable logging at runtime without modifying the application binary. The log4j package is designed so that these statements can remain in shipped code without incurring a heavy performance cost. Logging behavior can be controlled by editing a configuration file, without touching the application binary. One of the distinctive features of log4j is the notion of *inheritance* in loggers. Using a logger *hierarchy* it is possible to control which log statements are output at arbitrarily fine granularity but also great ease. This helps reduce the volume of logged output and minimize the cost of logging.
Ant 1.4.1	Ant est un outil du projet jakarta pour réaliser la compilation de projet java. C'est un équivalent à l'outils make sous Unix mais il est écrit en java et donc indépendant de toutes plateforme. Il permet donc la recompilation du projet sur toute plateforme équipée	Apache Jakarta Project	Ant utilise un fichier de paramètres (buildfile) pour la compilation du projet au format XML. Voir : http://ant.apache.org/manual/index.html

	d'un JVM.		
Junit 3.2 et JunitEE 1.3	Tests unitaires sur les EJB.	JUnit is Open Source Software, released under the IBM's Common Public License Version 1.0 and hosted on SourceForge : http://sourceforge.net/projects/junit/	Il est écrit en Java, mais des frameworks gratuits inspirés de celui-ci sont également disponibles pour de nombreux autres langages (C++, Python, etc.).
Jython 2.1	Scripts réalisant des tâches d'administration et de test. JPython est né d'une volonté d'utiliser les technologies Java avec un langage simple et puissant : Python. L'interpréteur python a donc été réécrit en Java pour permettre le mélange entre les classes du jdk et l'intrépeteur python au sein de la même machine virtuelle Java. Suite à des problèmes de license déténue par le CNRI (qui a vu la naissance de Python et de JPython), un nouveau nom a été donné à ce projet : Jython.	SourceForge : http://sourceforge.net/projects/junit/	Jython permet de : • compiler dynamiquement du python en bytecode Java • hériter des classes Java en Jython • compiler statiquement (création d'applets, servlets, beans...) • utiliser la syntaxe et les modules python dans les programmes Java • avoir un interpréteur python manipulant des objets Java • Jython est certifié 100% Pure Java (portabilité assurée)
PostgreSQL	Système de Gestion de Bases de Données Relationnel (SGBDR) fonctionnant sur des systèmes de type UNIX (par exemple Linux, FreeBSD, AIX, HP-UX, IRIX,Solaris, etc.). PostgreSQL est une base de données "Open Source". Le Groupe de Développement Global de PostgreSQL est une communauté de gens et de sociétés qui coopèrent pour faire avancer le développement de PostgreSQL, la base de données la plus sophistiquée du monde.	Groupe de Développement Global de PostgreSQL	PostgreSQL possède de nombreuses caractéristiques en faisant un SGBDR robuste et puissant digne des SGBD commerciaux: • des interfaces graphiques (X-Window est donc nécessaire) pour gérer les tables • des bibliothèques pour de nombreux langages (appelés *frontaux*) afin d'accèder aux enregistrements à partir de programmes écrits en Java (JDBC), C/C++, Perl, Tcl/Tk. • une API ODBC permettant à n'importe quelle application supportant ce type d'interface d'accèder à des bases de données de type PostgreSQL PostgreSQL fonctionne selon une architecture client/serveur, il est ainsi constitué : • d'une partie serveur, c'est-à-dire une application foncionnant sur la machine hébergeant la base de données (le serveur de bases de données) capable de traiter les requêtes des clients. Il s'agit dans le cas de PostgreSQL d'un programme résident en mémoire appelé *postmaster.* • d'une partie client devant être installée sur toutes les machines nécessitant d'accèder au serveur de base de données (un client peut éventuellement fonctionner sur

			le serveur lui-même) Les clients (les machines sur lesquelles le client PostgreSQL est installé) peuvent interroger le serveur de bases de données à l'aide de requêtes SQL.

[*] Logiciel propriétaire.
[1] Non applicable.

5. Scénario Pan-canadien

Pour l'élaboration d'un scénario de connexion pan-canadien, nous irons proposer l'impartition d'un centre de traitement de l'information à travers d'un fournisseur de services d'infrastructure de réseau et des données sans fil afin de fournir le lien entre les applications sans fil et les clients, les consommateurs, et les partenaires.

Le but de l'impartition dans ce cas spécifique est de rapidement permettre au service de technologie de l'information (TI) d'une société d'intégrer rapidement des services des messages sans fil dans ses applications et de faire de « tracking » sur des messages SMS générés par les dites applications.

Ainsi, dans les paragraphes qui suivent nous présenterons l'architecture nécessaire pour l'établissement d'un scénario pancanadienne et ensuite nous proposerons (et décrirons) les services de deux sociétés pour la sous-traitance : BelNexxia et MobileSys.

5.1. Belnexxia Inc.

5.1.1. Sommaire des Activités

Grâce au réseau de Bell Nexxia, les grandes entreprises ont accès à des solutions de communications complètes qui leur permettent d'étendre leur portée et de s'ouvrir sur de nouveaux horizons.

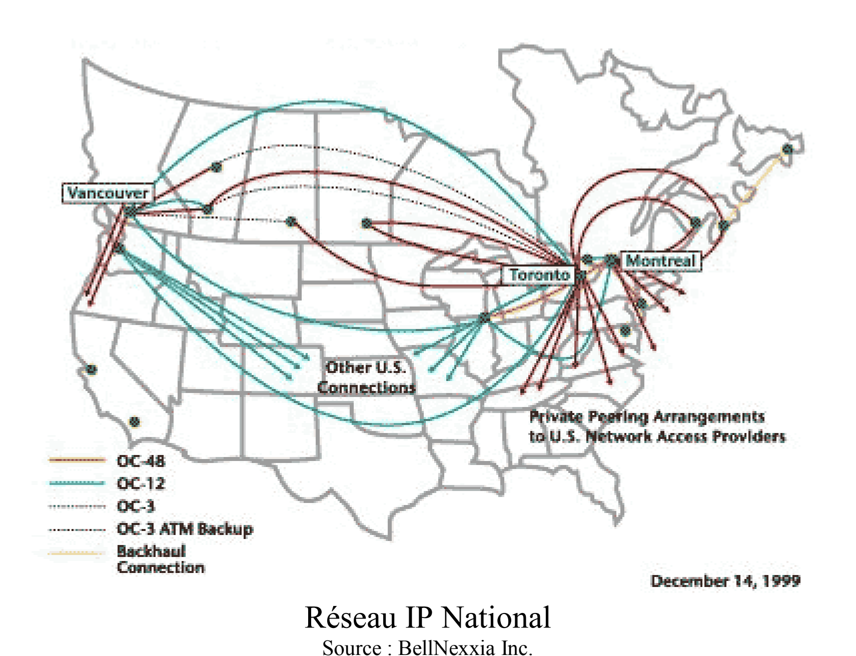
Réseau de transmission

Le réseau de transmission SONET/DWDM de Bell Nexxia repose sur notre infrastructure optique, composée de 12 fibres qui s'étendent d'est en ouest, dans toutes les grandes villes canadiennes et dans les principales métropoles américaines.

Réseau IP national

Pour répondre aux besoins émergents des services IP et Internet, Bell Nexxia et Cisco Systems se sont engagées dans une entente de co-développement unique afin d'établir et de faire évoluer un réseau IP transcanadien de nouvelle génération de haut niveau.

Pour répondre aux besoins émergents des services IP et Internet, Bell Nexxia et Cisco Systems se sont engagées dans une entente de co-développement unique afin d'établir et de faire évoluer un réseau IP transcanadien de nouvelle génération et haut niveau.



Réseau IP National
Source : BellNexxia Inc.

Le réseau IP de Bell Nexxia répond à un besoin criant en service Internet d'entreprise. Il donne aux entreprises accès à un réseau économique, sûr, fiable et étendu et pour lequel la qualité de service est garantie.

Bell Nexxia offre à ses clients un leadership technologique inégalé, de solides compétences en conception IP et les meilleurs services sous IP de l'industrie.
Architecture du réseau

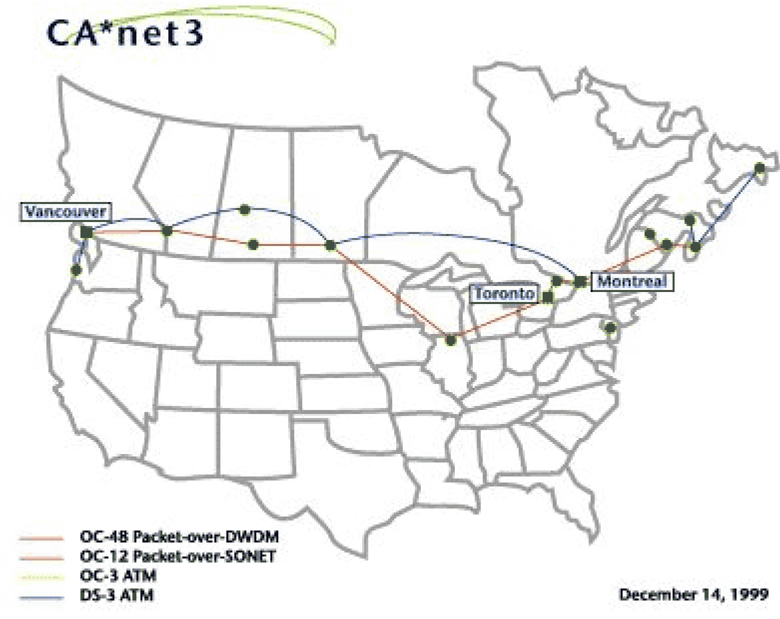
Le réseau IP utilise une architecture SONET de commutation de paquets, mode de transmission innovateur qui permet au routeur de giga-réseau de Cisco Systems de transmettre des paquets sur réseau IP à vitesse maximale par le truchement d'équipement SONET standard. Le réseau IP jouit d'une surviabilité complète, y compris une protection de 50 millisecondes.

Bell Nexxia offre en outre des fonctions de réseau privé virtuel (RPV) pour assurer l'exploitation d'applications sous IP évoluées, offrant ainsi aux entreprises les capacités d'un réseau privé dans une infrastructure de réseau public. Le réseau Bell Nexxia à solutions RPV intègre la sécurité à la couche 3

(réseau). L'environnement IP libre de connexion qui en résulte offre différentes classes de service et des fonctions de sécurité par groupes fermés d'usagers (GFU). L'architecture de commutation de paquets sur SONET remplace les services de couche 2 de l'ATM et du relais de trames, faisant ainsi gagner temps et argent aux clients.

Systèmes de soutien à l'exploitation

Grâce à son entente de co-développement avec Cisco Systems, Bell Nexxia aura accès aux plus récentes technologies en systèmes de soutien à l'exploitation (SSE) IP pour assurer la gestion rigoureuse de son réseau IP à grande échelle.



Systèmes de soutien à l'exploitation
Source : Bellnexxia Inc.

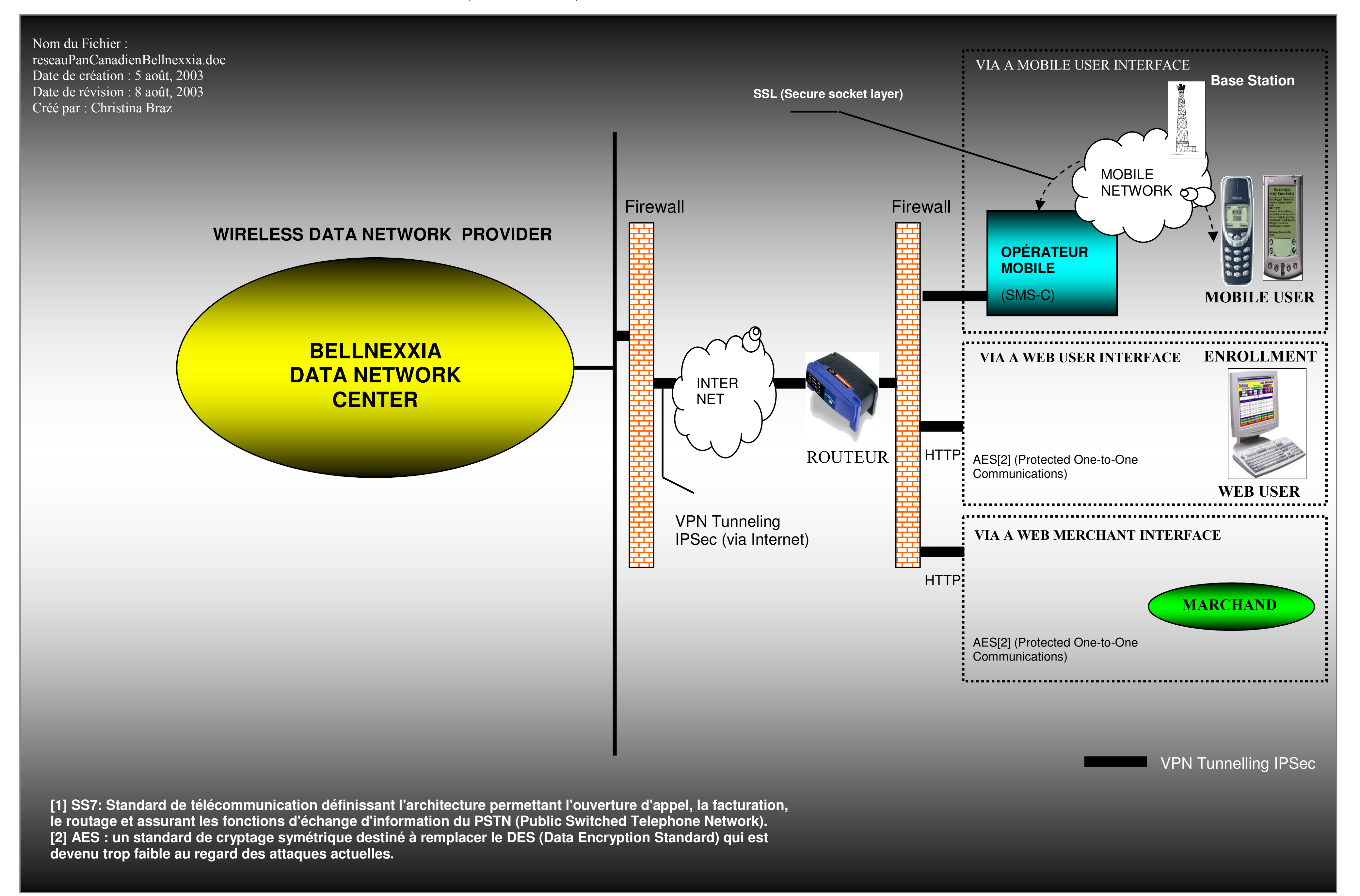
L'Internet de demain

En partenariat avec Canarie, Bell Nexxia a construit CA*net3, le réseau IP pan-canadien de prochaine génération pour la communauté scientifique. Son infrastructure optique, couplée à la technologie DWDM, permet une capacité allant jusqu'à 40 Gbps. Ça c'est de la largeur de bande ! Ensemble, Bell Nexxia et Canarie sont les pionniers de l'Internet de demain. Visitez le site Web de Canarie à l'adresse www.canarie.ca.

5.1.2. Architecture d'un Réseau Pan-canadien (Bellnexxia)

Voir la prochaine feuille.

Architecture d'un Réseau Pan-canadien (Bellnexxia)



6. Mobilesys Inc.

6.1. Sommaire des Activités

MobileSys est un fournisseur mondial d'infrastructure sans fil pour les entreprises et les sociétés tournées vers l'e-business. Son siège est situé à Mountain View en Californie et ses bureaux sont répartis aux Etats-Unis et en Europe. La solution de MobileSys permet la livraison des données sécurisée, fiable et indépendante des unités utilisées dans le monde entier via l'intégration de la technologie de "pousse" des données aux applications d'entreprise et l'interactivité vocale. MobileSys Network fournit aux clients une solution externalisée pour déployer, gérer et suivre les messages sans fil dans le monde entier. MobileSys compte plus de 700 clients dans le monde dont Cisco, Hewlett-Packard, Johnson and Johnson, LSI Logic, Qualcomm, Qwest, Stockpoint, Sun, et beaucoup d'autres.

MobileSys propose une solution globale comportant :

- un service dans le monde entier
- la garantie de transmission des messages
- un temps de latence prévisible
- la qualité du contrôle des services (QoS)
- des accords sur le niveau de service
- la sécurité

Les entreprises qui souhaitent externaliser le déploiement, l'administration et le suivi de ces messages pourront le faire via le réseau mondial de MobileSys (MobileSys Network).

En permettant l'intégration de MobileSys dans les infrastructures d'e-Business des sociétés, MobileSys les aide à profiter des avantages du réseau mondial, du suivi des messages de qualité de service et des solutions d'administration sans fil. Ainsi, les entreprises peuvent fournir des applications d'entreprise, des informations stratégiques et des ressources de messagerie à un large éventail d'utilisateurs mobiles en toute sécurité - partout et à tout moment.

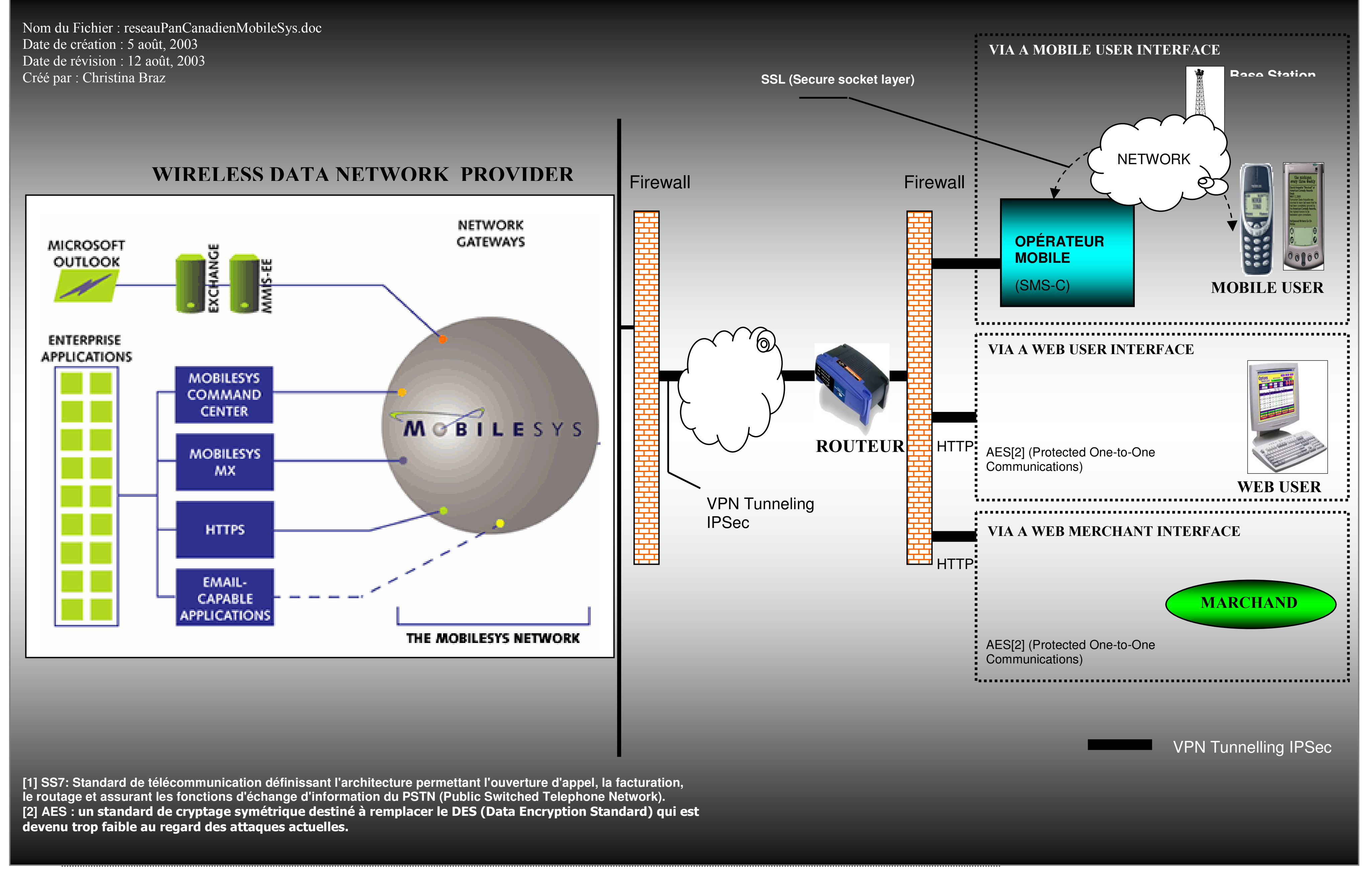
Le service d'infrastructure sans fil de MobileSys comprend la plate-forme de messagerie MX™ de MobileSys, qui simplifie l'intégration aux applications d'entreprise clés et le réseau de MobileSys (MobileSys Network), qui fournit des connexions gérées et fiables grâce aux porteuses sans fil du monde entier.

MobileSys Network offre aux entreprises, aux fournisseurs de services d'applications et aux entreprises tournées vers l'eBusiness une solution externalisée pour déployer, gérer et suivre les messages sans fil dans le monde entier. MobileSys Network exploite pleinement les relations avec les porteuses sans fil pour fournir une connexion fiable via quasiment tous les protocoles sans fil et unités activées par des données.

MobileSys Network est construit sur une architecture à tolérance de pannes pour garantir des couches supplémentaires de sécurité, le suivi des messages, la livraison fiable des messages et des temps de livraison prévisibles et mesurables. Les fonctions de suivi de la qualité de service de MobileSys permettent aux clients de CA de suivre chaque message de données sans fil de la diffusion à la réception.

Pour avoir plus d'informations sur MobileSys Inc., allez voir : http://www.mobilesys.com

6.2. Architecture d'un réseau pancanadien (Mobilesys Inc.)



7. Recommandations de Sécurité

Les réseaux mobiles ne supportent pas de la sécurité « end-to-end ». Pour cette raison, nous recommandons la mise en place d'un Virtual Private Network (VPN) pour la sécurité des trois scénarios proposés : Scénario Scénario Connexion directe, Réseau Cirano et Pan canadien.

7.1. Virtual Private Network (VPN)

Un VPN est une liaison sécurisée entre 2 réseaux distants à travers un réseau public, en général Internet. Cette technique assure l'authentification des 2 réseaux distants, l'intégrité des données et le chiffrage de celles-ci. Les 3 grands cas d'utilisation de VPN sont les suivants :

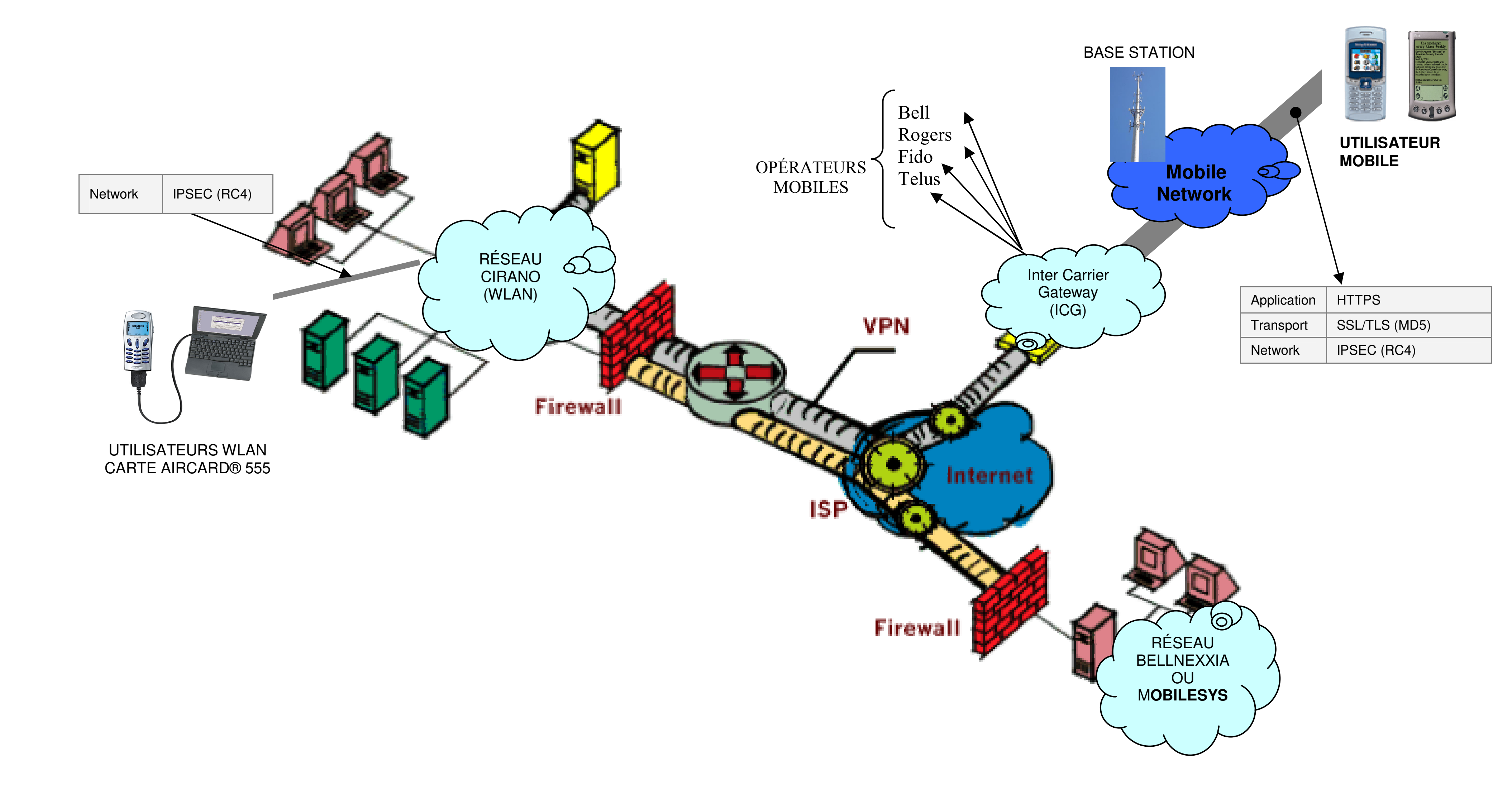
- Raccordement de télé travailleurs ou travailleurs mobiles. Ceux-ci se raccordent aux ressources de l'entreprise par modem ou tout autre moyen de connexion.
- Interconnexion de succursales. Des sites distants d'une même entreprise qui partagent les mêmes ressources sans avoir recours à des lignes spécialisées.
- Exploitation de réseaux extranets. Ce segment trouve sa justification dans l'essor du commerce électronique.

Toutes les communications sont chiffrées et toutes les connexions authentifiées (secret partagé ou certificat X509). Le standard IPSec (normalisé par l'IETF - Internet Engineering Task Force) permet la création de réseaux privés virtuels et les passerelles VPN permettent de créer ces réseaux de manière transparente pour les réseaux existants.

Voir ci-après la figure de l'architecture de sécurité.

Architecture de sécurité (Scénario Réseau Cirano et Pancanadien)

Nom du Fichier : architectureSecurite.doc
Date de création : 19 août, 2003
Date de révision : 26 août, 2003
Créé par : Christina Braz



7.2. IP Security Protocol (IPSec)

Protocole de sécurité au sein de la couche réseau. Ce protocole est développé pour fournir un service de sécurité à base de cryptographie, permettant de garantir l'authentification, l'intégrité, le contrôle d'accès et la confidentialité des données.

IPSec fournit les services suivants :

- Confidentialité des données : pas de décryptage possible au milieu du canal.
- Intégrité des données : pas de modification des données pendant le transport.
- Authentification de l'origine des données : check adresse IP source (non-répudiation).
- Anti-rejeu : pas de possibilité de rejouer des paquets afin de s'infiltrer dans une communication. Ceci est basé sur la vérification des numéros de séquences.

Mode tunnel (*Tunnelling*) :

Le mode tunnel correspond au cas ou au moins l'un des 2 peers IPSec se comporte comme une gateway IPSec. C'est à dire que le peer les décrypte/encrypte, mais les paquets ne lui sont pas directement destinés.

Exemple :

- Cas ou les 2 peers sont des gateways : si on fait un VPN LAN TO LAN (Local Area Network) => les deux peers encryptent et décryptent tour à tour, mais les paquets sont destinés aux LAN qu'elles protègent.
- Cas d'un poste nomade à une gateway

7.3. Rivest Cipher #4 (RC4)

Il s’agit d’une méthode de cryptographie à clé secrète développée par RSA Data Security Inc. pour authentifier et envoyer des certificats, et pour établir des clés. RC4 a un chiffrement de flux à clé de longueur variable, en général 40 ou 128 bits couplé à MD5, pour l'authentification des messages. RC4 est le plus rapide des modes de chiffrement.

7.4. Secure Socket Layer (SSL)/ Transport Layer Security (TLS)

SSL (Secure Socket Layer) est un protocole à négociation (on parle du « handshake » SSL), développé à l'origine par Netscape. Il a pour but de sécuriser les transactions Internet, par authentification du client (un navigateur la plupart du temps) et du serveur, et par chiffrement de la session.

La sécurisation des connexions à l'aide du protocole SSL doit assurer que :

- La connexion assure la confidentialité des données transmises
- La connexion assure que les données transmises sont intègres
- L'identité des correspondants peut être authentifiée
- La connexion est fiable

TLS (Transport Layer Security) protocol, développé par l'IETF, est la version 3.1 de SSL. SSL ne dépend pas des applications utilisées lors des transactions et s'applique sous les protocoles HTTP, FTP, Telnet, etc.

Clients et serveurs commencent par s'authentifier mutuellement, puis négocient une clé symétrique de session qui servira à assurer la confidentialité des transactions. L'intégrité de ces dernières est assurée par l'application de HMAC (Hashed Message Authentification Code).

7.5. MD5

C'est un algorithme créant une empreinte sur 128 bits pour un message donné en fonction de sa longueur arbitraire. L'algorithme est conçu de telle sorte qu'il est pratiquement impossible de produire deux messages ayant la même empreinte ou de produire un message ayant une empreinte spécifique. Cet algorithme propose une technique pour créer des signatures digitales et peut facilement s'associer à la technique de la clé privée/publique. L'algorithme MD5 serait vraisemblement développé par le MIT Laboratory for Computer Science et RSA Data Security. L'algorithme est encore en développement.

7.6. Secure Hypertext Transfer Protocol (HTTPS)

Protocole pour l´échange sécurisé de données sur le Web développé par Netscape. Protocole identique à HTTP mais sécurisé par cryptographie via SSL. HTTPS (S pour "secure" ou "sécurisé"). Les données sont cryptées, qu'il s'agisse de la requête ou de la page envoyée en réponse. Sur les sites marchands, au moment de saisir nos coordonnées bancaires, nous voyons apparaître https:// devant les adresses Web. Nos données sont cryptées avant d´être transmises.

3. Le cahier de charges



Le volet technologique MobiTicket

Cahier des Charges
MobiTicket

Projet d'enchère mobile (PEM)

Préparé par	Date	Work Package	Référence du document
Christina Braz	22-oct.-03	MobiTicket Version1.0	001

Classification du document	Nom du fichier	Institution	Ville/Province/Pays
Interne	CahierChargesMobiTicket.pdf	Centre interuniversitaire de recherche en analyse des organisations (CIRANO)	Montréal (Québec) Canada

Note :

Ce document est une version initiale des spécifications de conception du prototype de l'application appelée MobiTicket du Projet d'enchère mobile (PEM). En tant que tel, ce document ne prétend pas être une version finale de l'application mais plutôt être vu comme un point de départ pour des discussions et des travaux futurs.

TABLE DE MATIÈRES

1. Introduction

La création d'un prototype pour les Enchères VIP, nous permettra de présenter les fonctions principales que nous souhaitons implémenter, mais aussi d'obtenir des commentaires sur l'interface utilisateur (GUI). Nous le voyons comme un processus incrémental, hautement itératif, représentable par une boucle de rétroaction (feed-back).

Un cahier de charge est un document qui met en relief la spécification des besoins. Notre prototype MobiTicket phase I, comme tout produit ou logiciel a des spécifications et des besoins à satisfaire. Ces besoins et comment parvenir à mettre en place le prototype vont être détaillées tout au long de ce document.

Il est important souligner que nous aurons deux phases de développement de notre application dans le cadre du projet d'enchères mobile (PEM) :

- MobiTicket phase I : Ordinateur portable/Langage C++/Aircard 555.
- MobiTicket Phase II : Intégration Langage Java/GNP2/Base de données.

Le développement du prototype MobiTicket phase I respecte la notation Unified Modelling Language (UML).

1.1. But du projet

Le projet d'enchère mobile ("PEM) consiste à faire l'évaluation d'une opportunité de mise en marché de billets de spectacles en utilisant certaines fonctionnalités de la plate-forme de négociation générique (GNP2) de l'initiative "Towards Electronic Marketplaces" (TEM) du CIRANO. La particularité de ce projet est d'offrir la réservation des places en envoyant un prix d'offre à partir du téléphone sans fil des utilisateurs participants.

Pour cette première étape du projet, MobiTicket phase I, nous avons réalisé un prototype afin d'illustrer l'interaction de MobiTicket avec les utilisateurs et leurs téléphones mobiles. Il ne s'agit pas encore d'un produit à vocation commerciale mais d'un prototype pouvant entrer dans une démarche d'étude de faisabilité (MobiTicket phase II).

L'application se nommera "MobiTicket 1.0"[4].

[4] Le terme "mobiTicket" n'est pas enregistré comme marque de commerce au Canada (Source: Industrie Canada, 2003).

1.2. Liste des acteurs et leur mission

Maîtrise d'œuvre

Christina Braz
Maîtrise en commerce électronique/Université de Montréal-HEC Montréal
Responsable de la coordination du volet technologique

a. Processus de gestion des enchères (gestion de l'offre de service):

Objectifs:

- Définir les besoins et les caractéristiques d'accès (réseau, sécurité), d'exploitation (architecture du système) et les règles de gestion des informations nécessaire à la réalisation du processus d'enchère des billets disponibles à la vente (inventaire, paramétrage de GNP2, adjudication).

Livrables

- Définition des besoins et des caractéristiques de l'interface web.
- Programmation du site Web selon les besoins identifiés.
- Définition des besoins et des caractéristiques des messages SMS.
- Rédaction des règles de gestion des enchères et des principes d'adjutication.
- Définition des paramètres et des fonctions utilisées par GNP2.
- Modélisation du système de mise en enchère des billets.

b. Prototype

Objectifs:

- Réaliser l'application de mise en enchère de billets de spectacle.

Livrables:

- Un prototype fonctionnel du l'offre de service

2. Contexte

Partant du constat que nous sommes limités par le temps, nous allons définir des besoins qui se veulent cohérents par rapport à MobiTicket phase I. En outre, nous ne pouvons pas procéder à un test effectif de l'application MobiTicket 1.0 auprès d'utilisateurs réels (end-users), ou des utilisateurs indirects afin de jauger l'intérêt pour l'application.

2.1. Objectifs

L'application appelée MobiTicket a été développée dans le cadre du Projet d'enchère mobile (PEM). Le projet PEM mène des travaux sur la vente (réservation) de billets des spectacles sur un téléphone mobile. Il s'intéresse aux ventes aux enchères VIP et aux ventes à la dernière minute des billets des spectacles sur un téléphone mobile.

Le produit qui fait l'objet de ce cahier des charges correspond seulement à une implémentation d'une application de ventes aux enchères VIP. Notre but est de réaliser une application d'enchères mobile basée sur des messages SMS écrits dans le langage de programmation C++ afin de communiquer avec la carte réseau sans fil AirCard Sierra 555 installé sur un ordinateur portable.

2.2. Hypothèses

Liste des facteurs dont le changement conduirait à une révision du dossier de spécification des besoins logiciels:

- Intégration avec le GNP2 (Generic Negotiation Platform) et la base de données.
- Coexistence des langages de programmation C++ et Java afin de interagir avec GNP2 (achat d'un logiciel appelé "JunC++ion"[5] qui permet l'interopérabilité de ces langages).
- Interfaces Web.

Le produit MobiTicket phase I faisant l'objet de ce cahier des charges n'a pas encore une vocation commerciale.

[5] JunC++ion : C'est un outil de développement qui permet l'intégration des composants des langages de programmation C++ et Java. Ainsi, les deux langages sont capables de coexister dans un processus sans CORBA ou JINI. Voir : http://www.codemesh.com/en/CodemeschWHitepaper.pdf

3. Description générale

3.1. Intégration dans le système

Le produit est réalisé de manière indépendante des autres modules tels que les interfaces Web pour l'inscription et pour la configuration des enchères VIP par le marchand, le GNP2 et la base de données. Il n'y a donc pas pour le moment d'interaction avec ces modules.

Cependant, le projet pourra s'inscrire dans une démarche pour la réalisation d'un projet-pilote dans le cadre du projet PEM. Son but est de présenter les possibilités de négociation au sein d'environnement GNP2 à travers d'une application mobile (MobiTicket). Nous entendons par cela permettre l'accès aux ventes aux enchères VIP et à la dernière minute des billets des spectacles depuis tout le téléphone mobile, et ceci de manière totalement transparente pour l'utilisateur mobile.

3.2. Fonctions du projet

3.2.1. MobiTicket Phase I (Ordinateur portable/C++/Aircard 555) :

Le produit doit pouvoir communiquer les enchères VIP à travers des messages Push SMS[6] sur le téléphone mobile d'utilisateur. La Figure 1 démontre les échanges de messages entre l'utilisateur mobile et le Marchand.

L'utilisateur mobile recevra initialement une annonce d'une vente aux enchères VIP par le marchand et selon ses saisies recevra les messages suivants:

- d'acquittement pour une mise valide.
- des erreurs pour les mises non-valides.
- de l'état d'avancement d'enchère dans le cas où l'utilisateur mobile remporte ou pas l'enchère.
- de clôture de l'enchère.

Il est important de souligner que nous avons utilisé seulement deux téléphones mobiles afin de démontrer l'interaction de la communication mobile.

Le prototype pour cette phase est déjà finalisé et fonctionnel (démo).

[6] Service Push SMS: c'est un service mobile de télécommunications permettant de fournir, après une activation réussie du service par les utilisateurs mobiles eux-mêmes, des informations payantes sur un thème demandé au moyen de SMS. Les utilisateurs peuvent par exemple s'informer sur les actualités, les prévisions météorologiques, la bourse, etc.

Figure 1

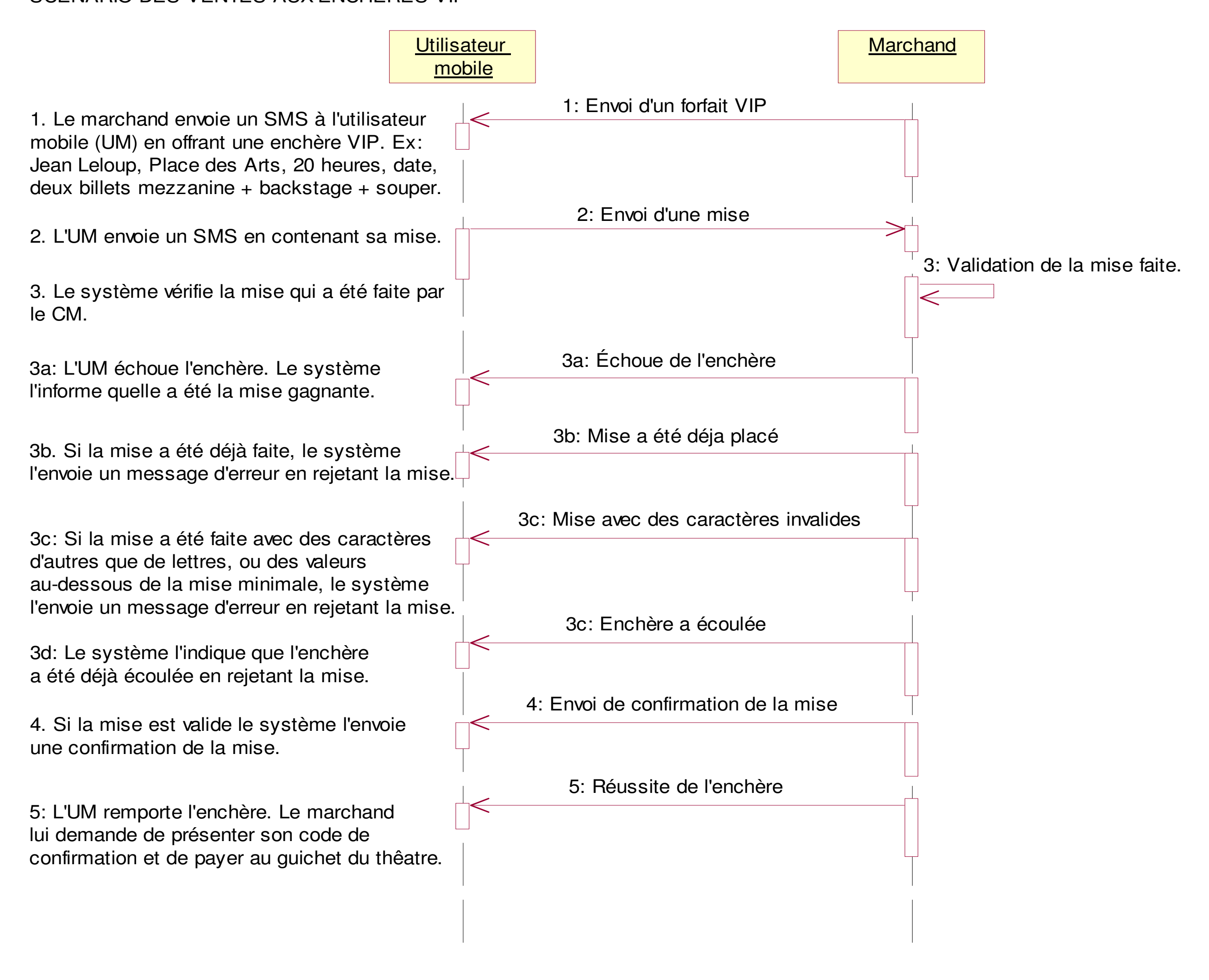


Diagramme de séquence de l’application "Enchères VIP".
Source : Braz, C., 2003.

3.2.2. MobiTicket Phase II (GNP2) :

Le produit doit pouvoir communiquer les ventes aux enchères VIP ou les ventes à la dernière minute à travers des messages Push SMS sur le téléphone mobile d'utilisateur.

Les échanges de messages pour les enchères VIP entre l'utilisateur mobile, le programme Java "Traducteur SMS>GNP" et bien le GNP2 sont démontrées dans la Figure 2.

Figure 2

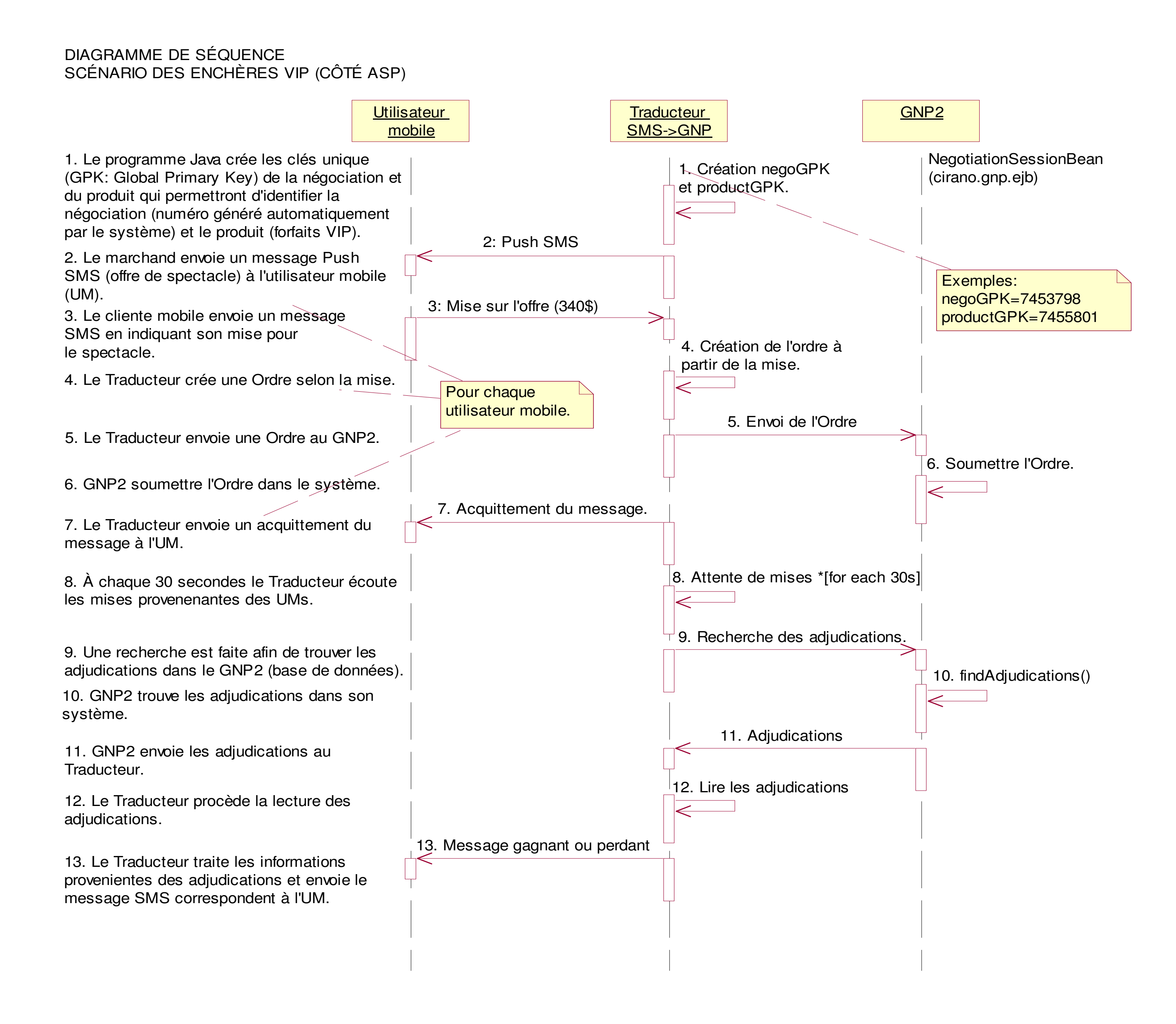


Diagramme de séquence de l'application "Enchères VIP (Côté ASP)".
Source : Braz, C. & Therrien, I. (CIRANO, 2003).

La Figure 3 démontre les échanges de messages pour les ventes à la dernière minute entre l'utilisateur mobile et le marchand (Exemple : Place des Arts).

Figure 3

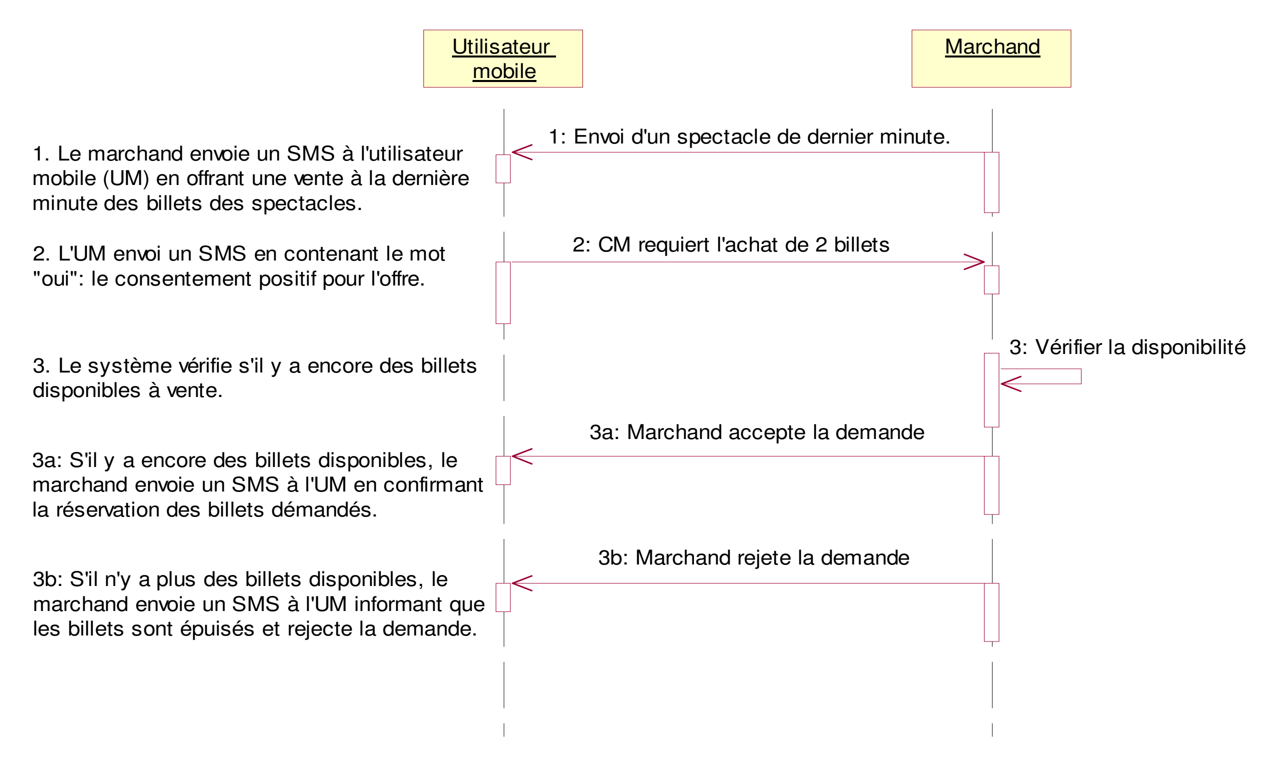


Diagramme de séquence de l'application "Dernier minute".
Source : Braz, C. (CIRANO, 2003).

Pour cette phase II, nous utiliserons deux ou plusieurs téléphones mobiles afin de démontrer l'interaction de la communication mobile.

Le prototype pour cette phase n'est pas encore finalisé. Prévision : 2003-12-31.

3.3. Caractéristiques des utilisateurs

Les utilisateurs potentiels de MobiTicket phase I devront être capables de manipuler les interfaces de l'application en langage C++ afin de pouvoir envoyer et recevoir des messages des numéros de téléphones mobiles.

3.4. Contraintes de développement

3.4.1. Moyens matériels

Un ordinateur portable avec :

- Trousse de données sans fil incluant la carte Sierra Wireless Aircard 555 CDMA 1X (carte réseau sans fil), le logiciel Watcher et l'antenne.
- Forfait de données Réseau 1X (Opérateur de télécommunications : Bell Mobilité).

3.4.2. Moyens logiciels

- L'environnement de développement Microsoft Visual C++ 6.0
- Librairie MSDN (Microsoft Developper Network)

3.4.3. Moyens financiers

Le MobiTicket phase I n'a pas une vocation commerciale. Les moyens disponibles sont ceux internes au CIRANO, aucun budget n'est disponible pour l'acquisition d'une solution commerciale.

4. Expression détaillée des requis

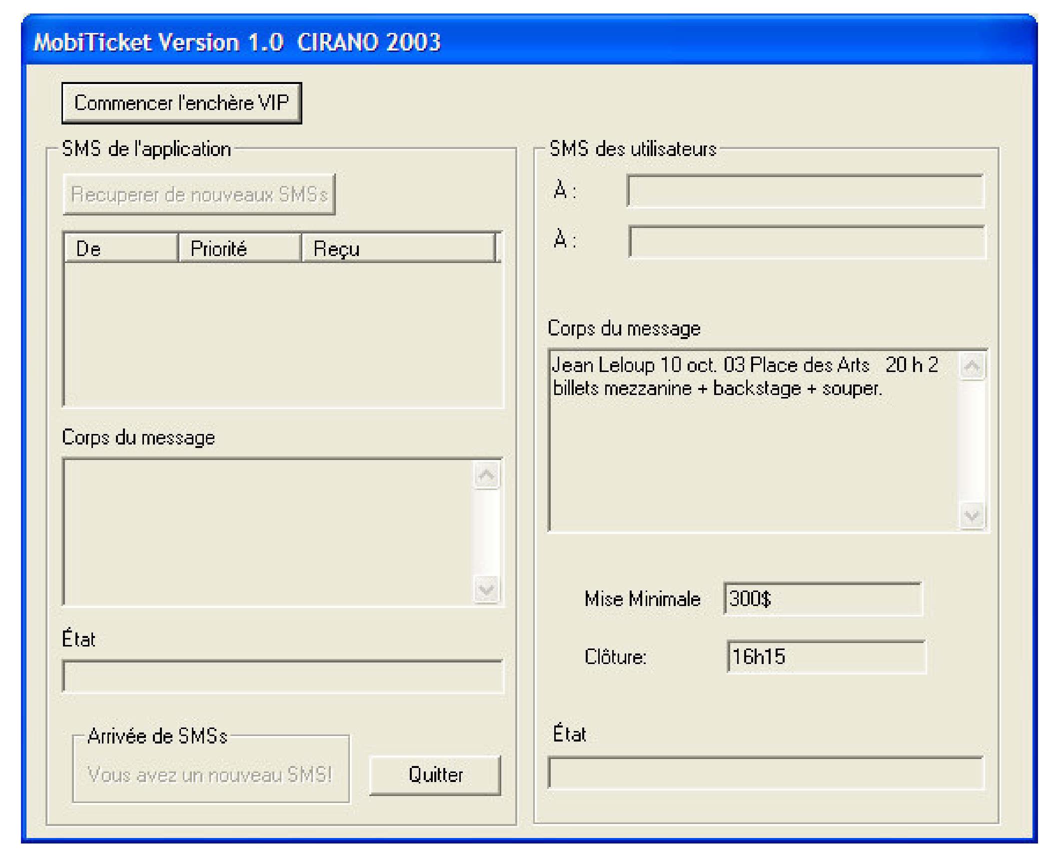
4.1. Spécifications fonctionnelles

Les spécifications fonctionnelles décrivent les fonctions (ou les opérations) que le logiciel doit réaliser. L'application MobiTicket phase I prend en entrée une "interface de configuration des "enchères VIP" selon les Figure 4, 5 et 6 qui sera manipulée par l'utilisateur de l'application (ASP) obtenue localement sur l'ordinateur portable à travers l'exécution du programme C++ installé sur cette machine. Voir les détails des matériels dans la section 4.3.

4.2. Spécifications d'interface

Aucune base de données n'est mise en jeu dans cette application (MobiTicket phase I). En outre, la carte réseau doit être placé dans l'ordinateur portable mais sans la connexion active (1XRtt[7]) afin que les interactions entre l'application et les téléphones mobiles puissent être réalisées.

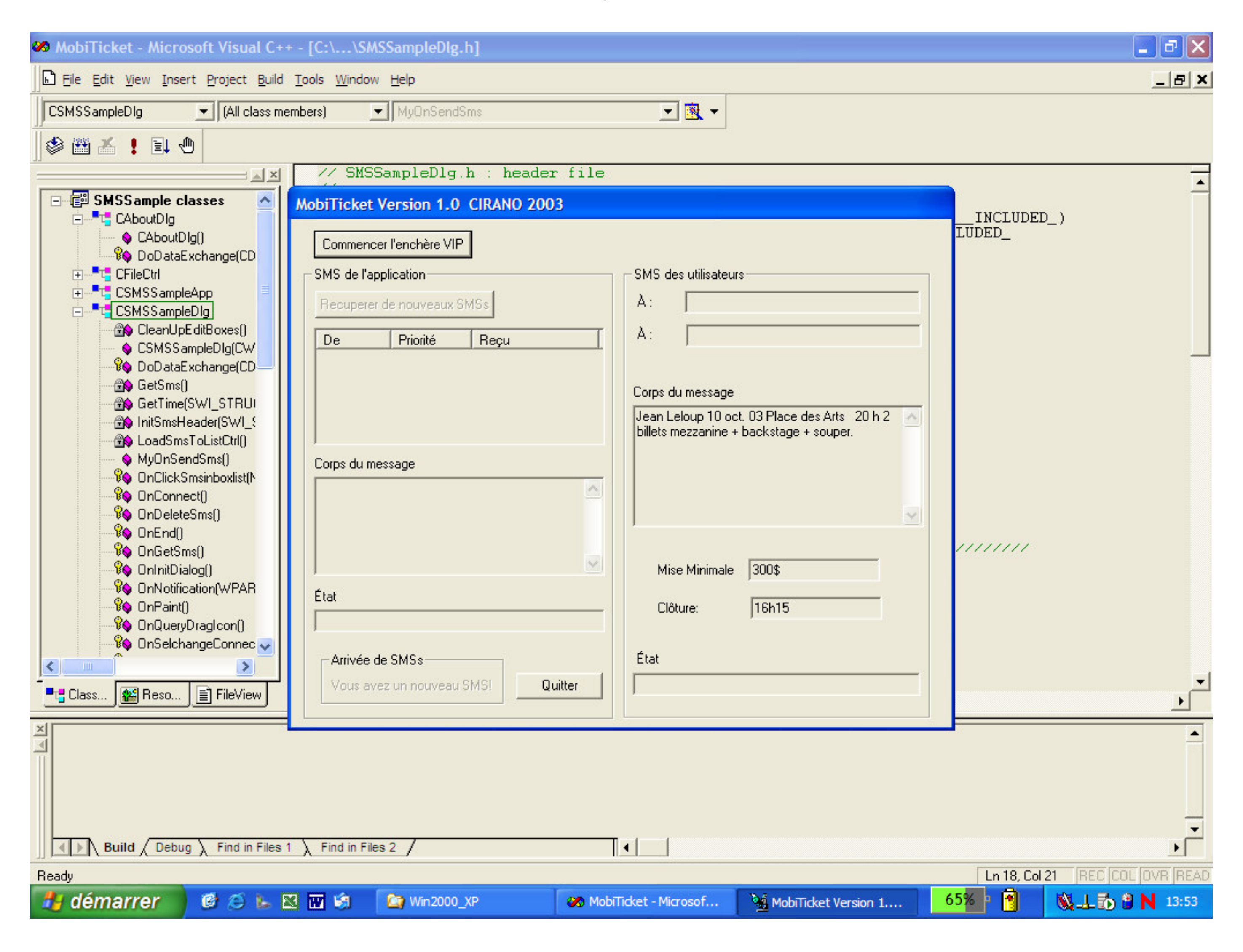
Figure 4



Interface de configuration d'enchère VIP (MobiTicket phase I).
Source : Braz, C. (CIRANO, 2003).

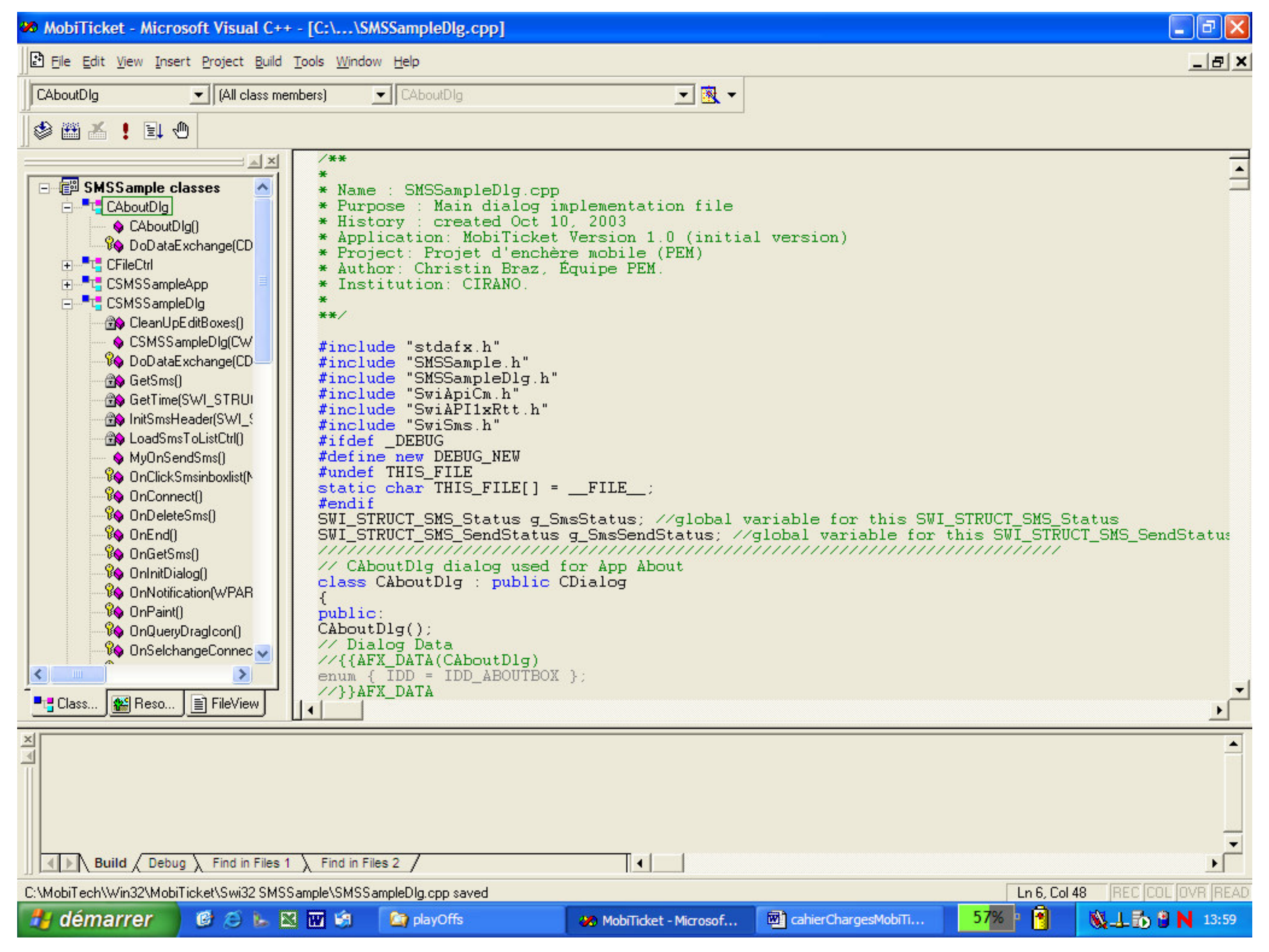
[7] La technologie réseau 1xRTT permet d'atteindre des vitesses de transmission de 144 Kbps.

Figure 5



Interface de configuration d'enchère VIP étant éxecutée sur l'environnement de développement Microsoft Visual C++ 6.0 (MobiTicket phase I).
Source : Braz, C. (CIRANO, 2003).

Figure 6



L'environnement de développement Microsoft Visual C++ 6.0 (MobiTicket phase I).
Source : Braz, C. (CIRANO, 2003).

4.2.1. Interaction entre MobiTicket et l'utilisateur mobile

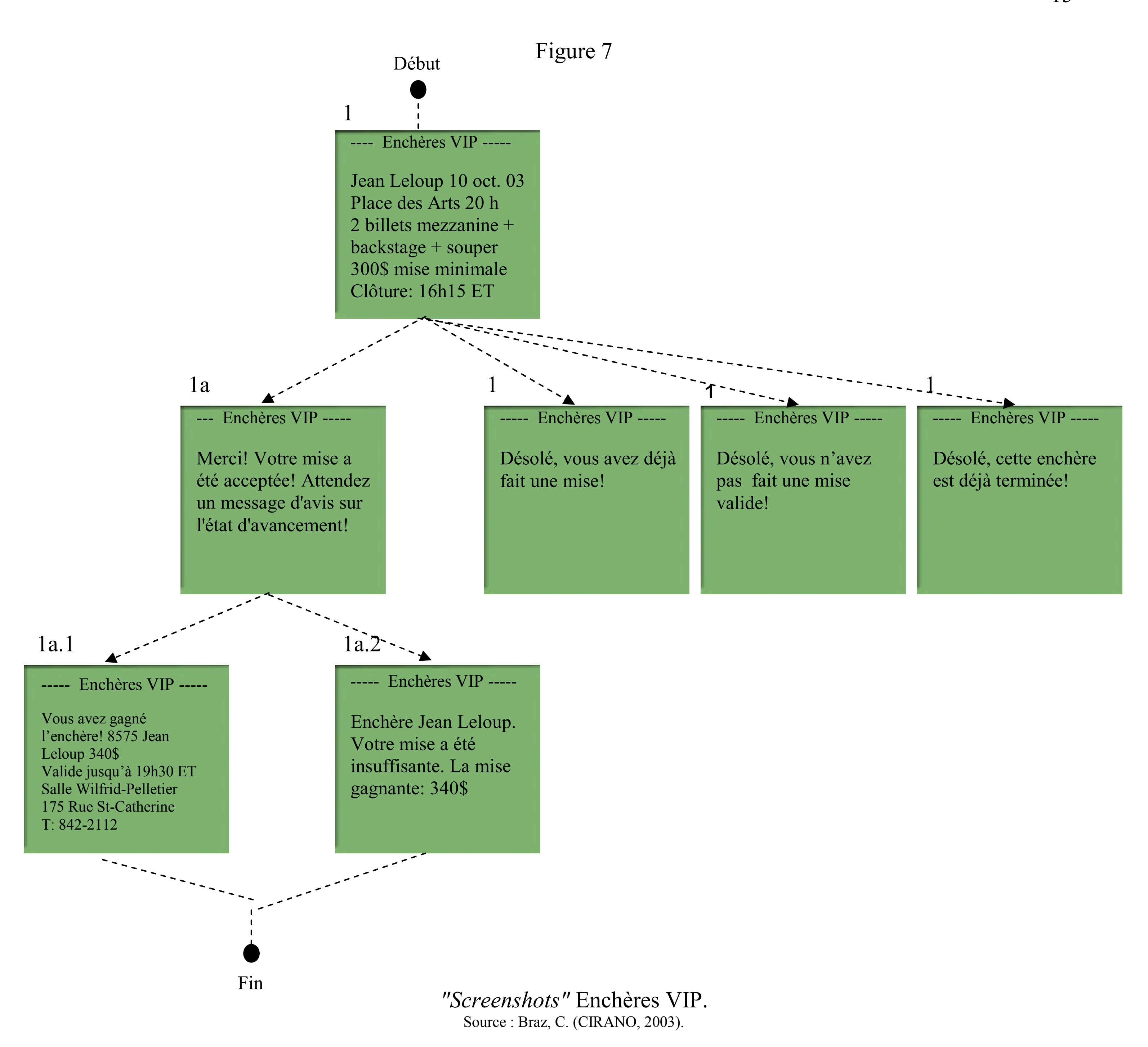
Nous présentons ci-dessous les messages échangés entre l'application MobiTicket / l'utilisateur mobile et les *"screenshots"* dans la Figure 7 :

Description de l'enchère	Annonce d'une vente aux enchères VIP (2 billets + offres promotionnelles) par le marchand sur un téléphone mobile : *Jean Leloup 10 oct. 03* *Place des Arts 20 h* *2 billets mezzanine + backstage + souper* *300$ mise minimale Clôture: 16h15 ET*
Règles de adjudication	L'utilisateur mobile[8] (UM) recevra des messages SMS étant dans l'un des cas suivants : 1a) Si l'UM a fait une mise valide (SMS d'acquittement): *Votre mise a été acceptée! Attendez un message d'avis sur l'état d'avancement!* 1a.1) Si l'UM remporte l'enchère : *Vous avez gagné l'enchère! 8575 Jean Leloup 340$* *Valide jusqu'à 19h30 ET Salle Wilfrid-Pelletier* *175 Rue St-Catherine T : 842-2112* Note : 8575 Jean Leloup 340$ (Code du spectacle + Nom du spectacle + Valeur misée). 1a.2.) Si l'UM perds l'enchère : *Enchère Jean Leloup. Votre mise a été insuffisante.* *La mise gagnante: 340$* 1b.) Si l'UM a misé plus d'une fois : *Désolé, vous avez déjà fait une mise!* 1c.) Si l'UM a misé une valeur inférieure ou égal à la mise minimale, ou a saisit des caractères autres que des lettres : *Désolé, vous n'avez pas fait une mise valide!* 1d.) Si l'UM a misé après l'heure de la fin d'enchère : *Désolé, cette enchère est déjà terminée!*

[8] Utilisateur mobile : le détenteur d'un téléphone mobile en étant abonné au service d'un opérateur mobile.

15

Figure 7



"Screenshots" Enchères VIP.
Source : Braz, C. (CIRANO, 2003).

16

4.3. Architecture de l'application MobiTicket

L'architecture de l'application MobiTicket est constituée de trois modules principaux : Module Client, Module AirCard et le module GNP2.

4.3.1. Module Client (programme C++)

a. Matériel :

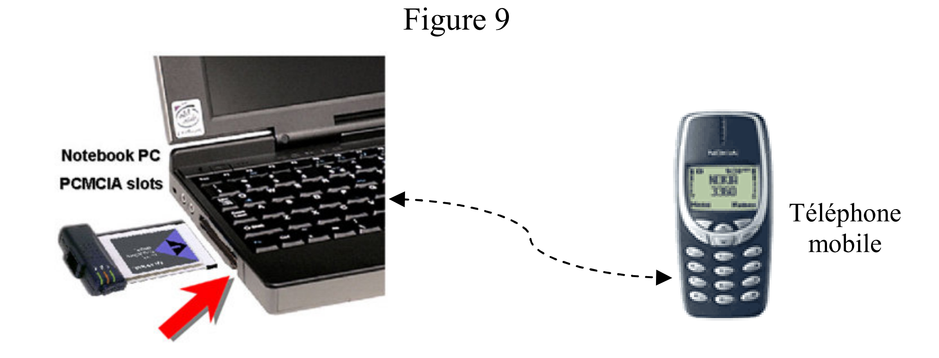
Le Module client interagit avec le Module AirCard à travers d'une interface de configuration d'enchère VIP (Figure 4) conçue en langage de programme C++ qui permet d'envoyer et de recevoir des messages SMS aux utilisateurs mobiles.

La carte réseau sans fil AirCard 555 s'insère dans un réceptacle standard pour carte PC de type II (présent sur la plupart des ordinateurs bloc-notes) et joue le rôle d'une carte réseau, d'un modem et d'un téléphone sans fil (Figures 8 et 9). Sans câble réseau ni ligne téléphonique, elle permet de se connecter à Internet ou à un réseau d'entreprise, d'envoyer ou de recevoir des messages SMS, du courriel et d'émettre des appels téléphoniques. L'AirCard 555 fonctionne via un type de réseau sans fil appelé CDMA (Code Division Multiple Access, accès multiple à répartition par code). Cette technologie offre, outre la liaison sans fil, la fonctionnalité de SMS (Short Messaging Service), qui permet d'envoyer et de recevoir des mini-messages via l'AirCard. La trousse de la carte AirCard 555 contient les éléments suivants : Carte réseau sans fil AirCardMD 555, Antenne, CD d'installation contenant le logiciel de l'AirCard. L'AirCard est fournie avec les logiciels suivants : Une application appelée "Watcher", destinée à gérer l'AirCard et à surveiller nos connexions, une application appelée Gestionnaire de carte réseau, qui permet de choisir entre l'AirCard 555 et d'autres cartes réseau, et le pilote du logiciel qui crée l'interface entre la carte réseau et notre système d'exploitation Windows.

Figure 8

Carte réseau sans fil AirCard® 555 de Sierra Wireless.
Source: Sierra Wireless Inc., 2003.

Figure 9



L'AirCard® 555 nous permet d'accéder à nos données en glissant la carte dans l'emplacement pour carte PC de notre ordinateur portatif ou de poche.
Source: Sierra Wireless Inc., 2003.

Pour pouvoir utiliser l'AirCard 555, nous devons disposer d'un compte de transmission de données sans fil 1X (Bell Mobilité) nous permettant d'accéder à un réseau CDMA. L'AirCard fonctionne avec la nouvelle technologie réseau 1xRTT. La technologie One times (1X) est l'évolution 3G de la technologie AMRC (accès multiple à répartition par code), la norme privilégiée pour les télécommunications sans fil en Amérique du Nord. 1X est une technologie à haute vitesse efficace qui double les capacités de transmission de la voix et la vitesse de transmission des données par paquet et qui prolonge la durée des piles de téléphones.

b. Logiciel :

EXEMPLE DE CODE POUR L'ENVOI DES MESSAGES SMS (FONCTION ONSENDSMS) :

```
/**
*
* Function : OnSendSms
* Description : Handler for Send SMS control button
* Arguments : None
* Returns : None
*
**/

void CSMSSampleDlg::OnSendSms(CString message)
{
// Add my control notification handler code here
SWI_STRUCT_SMS_StoreSms sSms;
BOOL bInterimEmailSolution = TRUE; // for now this one is always on
CString szMsgBody;
CString szAddress;
CString szAddress2;
CString szTemp;

if (message == "")
{
GetDlgItem(IDC_COMPOSEMSGBODY)->GetWindowText(szMsgBody);
if (!szMsgBody.GetLength())
{
MessageBox("Pas capable d'envoyer de message vide !");
return;
}
szMsgBody+=" Mise minimale: ";
CString s;
GetDlgItem(IDC_SENDSMSTO_BIDMIN)->GetWindowText(s);
szMsgBody+=s;
szMsgBody+=" Cloture:";
GetDlgItem(IDC_SENDSMSTO_CLOT)->GetWindowText(s);
szMsgBody+=s;

}
else
{
szMsgBody=message;
}
if (!szMsgBody.GetLength())
{
MessageBox("Pas capable d'envoyer un message vide !");
return;
```

```
}
//check for message length
if (BUFFER_SIZE < szMsgBody.GetLength())
{
MessageBox(_T("Le message ne peut pas excéder 200 octets"), NULL, MB_OK);
return;
}
// initialize SMS header
InitSmsHeader(&sSms.sSMSHeader);
// get TO address
GetDlgItem(IDC_SENDSMSTO)->GetWindowText(szAddress);

// customer1=szAddress;
if ( szAddress.GetLength() )
{
                // set destination address
                for (I32 i = 0; i < szAddress.GetLength(); i++)
                {
                sSms.sSMSHeader.szAddress[i] = (U8)szAddress.GetAt(i);
                }
                sSms.sSMSHeader.addrLen = szAddress.GetLength();
                sSms.sSMSHeader.addrDigitMode = 0;
                sSms.sSMSHeader.addrNumMode = 0;
                sSms.sSMSHeader.addrNumType = 0;
                sSms.sizeSMSBody = szMsgBody.GetLength();
                sSms.pMsgBody = (U8*)szMsgBody.GetBuffer(sSms.sizeSMSBody);
                // random MsgId in range 0-50 - these limits are arbitrary
                sSms.nSMSId = rand();
                if (sSms.nSMSId > 50)
                {
                sSms.nSMSId = sSms.nSMSId % 50;
                }
                // store the message to the modem
                rCode = SwiStoreSMSMessage(&sSms, DEFAULT_TIMEOUT);

                if (SWI_RCODE_OK != rCode)
                {
                ProcessError(rCode);
                return;
                }

}



SWI_STRUCT_SMS_StoreSms sSms2;

// initialize SMS header
InitSmsHeader(&sSms2.sSMSHeader);
// get TO address
GetDlgItem(IDC_SENDSMSTO_SND)->GetWindowText(szAddress2);
// customer2=szAddress;
if ( (!szAddress2.GetLength()) && (!szAddress.GetLength()) )
                {
                MessageBox(_T("Les adresses sont vides!"), NULL, MB_OK);
                return;
                }
if ( szAddress2.GetLength())
{
                // set destination address
for (I32 i = 0; i < szAddress2.GetLength(); i++)
```

```
{
sSms2.sSMSHeader.szAddress[i] = (U8)szAddress2.GetAt(i);
}
sSms2.sSMSHeader.addrLen = szAddress2.GetLength();
sSms2.sSMSHeader.addrDigitMode = 0;
sSms2.sSMSHeader.addrNumMode = 0;
sSms2.sSMSHeader.addrNumType = 0;
sSms2.sizeSMSBody = szMsgBody.GetLength();
sSms2.pMsgBody = (U8*)szMsgBody.GetBuffer(sSms2.sizeSMSBody);
// random MsgId in range 0-50 - these limits are arbitrary
sSms2.nSMSId = rand();
if (sSms2.nSMSId > 50)
{
sSms2.nSMSId = sSms2.nSMSId % 50;
}
// store the message to the modem
rCode = SwiStoreSMSMessage(&sSms2, DEFAULT_TIMEOUT);

if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
return;
}

}

// send the message
rCode = SwiSendSMS(DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
return;
}

GetDlgItem(IDC_SENDSMS)->EnableWindow(FALSE);
}
```

EXEMPLE DE CODE POUR LA RÉCEPTION DES MESSAGES SMS :

```
/**
*
* Function : GetSms
* Description : Function to get received SMS
* Arguments : None
* Returns : None
*
**/

void CSMSSampleDlg::GetSms()
{
MessageBox("Récupérer SMSs");

SWI_STRUCT_SMS_Status sSmsStatus;
SWI_STRUCT_SMS_RetrieveSms sSms;
U32 msgCount;
CString szTemp;
TCHAR szMessage[SWI_MAX_BUFFER_SIZE];
TCHAR szTime[SWI_MAX_TIME_SIZE];
GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T(""));
```

```
// any new messages
sSmsStatus.sizeStruct = sizeof(SWI_STRUCT_SMS_Status);
rCode = SwiGetSMSMessageStatus(&sSmsStatus, DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
      {
      ProcessError(rCode);
      return;
      }
//Find out how many messages are stored in the modem
msgCount = sSmsStatus.cntRegularMsg + sSmsStatus.cntUrgentMsg + sSmsStatus.cntVoiceMsg;
if (!msgCount)
      {
      GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T("Pas de messages a lire"));
      LoadSmsToListCtrl();
      if (mesg1 !="")
            {OnSendTargetSms(mesg1,adr1);mesg1="";}
      if (mesg2 !="")
      {OnSendTargetSms(mesg2,adr2);mesg2="";}

      return;
      }
// allocate the buffer for the message
sSms.sizeStruct = sizeof(SWI_STRUCT_SMS_RetrieveSms);
sSms.sizeBuffer = BUFFER_SIZE;
sSms.pMessage = new U8[BUFFER_SIZE];
// retrieve the messages
TCHAR szAddress1[MAX_SMSI_TL_ADDRESS];
TCHAR szAddress2[MAX_SMSI_TL_ADDRESS];
TCHAR szTempTime[MAX_TIME_SIZE];
BOOL different=FALSE;
I32 i;
SWI_TYPE_SMS_Priority ePriority = SWI_SMS_PRIORITY_NORMAL;

int bid;
int iter ;
m_File.getInfo(0, szAddress1, &ePriority, szTempTime);
m_File.getInfo(1, szAddress2, &ePriority, szTempTime);
while (msgCount > 0)
      {
      rCode = SwiRetrieveSMSMessage(&sSms, DEFAULT_TIMEOUT);
      msgCount = sSms.cntRemainingSMS;
      GetDlgItem(IDC_NEWSMS_NOTIF)->EnableWindow(FALSE);
      if (SWI_RCODE_OK != rCode)
            {
            ProcessError(rCode);
            return;
            }
      //extract the time and the Message to character arrays
      GetTime(sSms, szTime);
      UpdateSmsOutbox(sSms.pMessage, szMessage);
      if (my_bIsConnected == FALSE)
            {
            if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                              {
                              adr1[i]=sSms.sHeader.szAddress[i];
                              }
            mesg1="Desole, cet enchere est deja terminee!";
            //OnSendTargetSms("Desole,        cet        enchere        est        deja
terminee!",sSms.sHeader.szAddress);
```

```
            }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Desole, cet enchere est deja terminee!";
            }

            }
      else
            {
            for (i = 0; (i < 11) ; i++)
                  {
                  if (sSms.sHeader.szAddress[i] != (U8)szAddress1[i] )
                        different= TRUE;
                  }
            if (different==false)
                              {
                  if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Desole, vous avez deja fait une mise!";
            }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Desole, vous avez deja fait une mise!!";
            }

            //OnSendTargetSms("Desole,      vous      avez      deja      fait      une
mise!",sSms.sHeader.szAddress);
            }


            else
                  {
                  for (i = 0; (i < 11); i++)
                        {
                        if (sSms.sHeader.szAddress[i] != (U8)szAddress2[i] )
                        different= TRUE;
                        }

            if (different==false)
{
if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Desole, cet enchere est deja terminee!";
            //OnSendTargetSms("Desole,        cet        enchere        est        deja
terminee!",sSms.sHeader.szAddress);
```

```
            }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Desole, cet enchere est deja terminee!";
            }
}

                  //OnSendTargetSms("Desole,     vous     avez     deja     fait     une
mise!",sSms.sHeader.szAddress);
            else
                  {
                  // New bid to handle auction

                  CString s = CString (szMessage);

                  for (iter=0;s.GetAt(iter)==' ';iter++)
                        {;}
                  if (s.GetAt(iter)=='$')
                        {iter++;}
                  for (;s.GetAt(iter)==' ';iter++)
                        {;}
                  bid=0;
                  for (; iter < s.GetLength(); iter++)
                        {
                        switch (s.GetAt(iter))
                              {
                              case  '0' :
                              bid=(bid*10);
                              break;
                              case  '1' :
                              bid=(bid*10)+1;
                              break;
                              case  '2' :
                              bid=(bid*10)+2;
                              break;
                              case  '3' :
                              bid=(bid*10)+3;
                              break;
                              case  '4':
                              bid=(bid*10)+4;
                              break;
                              case  '5':
                              bid=(bid*10)+5;
                              break;
                              case  '6' :
                              bid=(bid*10)+6;
                              break;
                              case  '7':
                              bid=(bid*10)+7;
                              break;
                              case  '8':
                              bid=(bid*10)+8;
                              break;
                              case  '9':
                              bid=(bid*10)+9;
                              break;
                              default :
```

```
                              iter = s.GetLength();
                              break;
                              }
                        }

                  if (bid <= bidmin)
                        {

if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Desole, vous n'avez pas  fait une mise valide!";
            //OnSendTargetSms("Desole,        cet        enchere        est        deja
terminee!",sSms.sHeader.szAddress);
            }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Desole, vous n'avez pas fait une mise valide!";
            }



                        //MessageBox("bid non valide");
                        //OnSendTargetSms("Desole,  vous  n'avez  pas   fait  une  mise
valide!",sSms.sHeader.szAddress);
                        }
                  else
                        {
                        if (bid1==0)
                              {
                              bid1=bid;
                              for (i = 0; (i < 12) ; i++)
                                    {
                                    num1[i]=sSms.sHeader.szAddress[i];
                                    }
                              //MessageBox("first        if        ");MessageBox(num1);
      MessageBox(num2);
                              }
                        else
                              {
                              bid2=bid;
                              for (i = 0; (i < 12) ; i++)
                                    {
                                    num2[i]=sSms.sHeader.szAddress[i];
                                    }
                              //MessageBox("first       else       ");MessageBox(num1);
      MessageBox(num2);
                              }
                        CString bids = CString (bid);
                        m_File.addToFile(sSms.sHeader.szAddress,
sSms.sHeader.ePriority,szTime, szMessage);
                        if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
```

```
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Merci! Votre mise a ete acceptee! Attendez un message d'avis sur
l'etat d'avancement!";
                  }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Merci! Votre mise a ete acceptee! Attendez un message d'avis sur
l'etat d'avancement!";
            }

                        //OnSendTargetSms("Merci! Votre mise a ete acceptee! Veuillez
patienter et attendre une reponse! ",sSms.sHeader.szAddress);
                        }

                  }
            }
      }

      }

      if (m_SmsInboxList.GetItemCount())
      {
            // set the selected item in list control to have index 0 (arbitrary)
            //
            m_SmsInboxList.SetItemState(0, LVIS_SELECTED, LVIS_SELECTED);
                        }

      m_SmsInboxList.SetItemState(0, LVIS_SELECTED, LVIS_SELECTED);
      GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T("SMS a ete recupere avec
succes"));

      //update the list box
      LoadSmsToListCtrl();

      // free the memory
      delete [] sSms.pMessage;


      UpdateData(FALSE);
}
```

4.3.2. Module AirCard (carte réseau sans fil)

Le Module AirCard interagit avec le Module client à travers d'une interface de configuration d'enchère VIP (Figures 4 et 10) conçue en langage de programme C++ qui permet d'envoyer et de recevoir des messages SMS aux utilisateurs mobiles.

4.3.3. Module GNP2

Ce module fait partie de la phase II où nous irons établir l'Intégration avec le GNP2 et la base de données installé sur le serveur "Toja" à partir d'un programme Java qui parlera à GNP2 (Figure 10). Il faudra transcrire le programme fait en C++ en un programme Java.

Figure 10

MOBITICKET PHASE I
MOBITICKET PHASE II
MODULE CLIENT
Programme C++
MODULE AIRCARD
External Java API
MODULE GNP2
BASE DE DONNÈES
TOJA (CIRANO)
ORDINATEUR PORTABLE
Envoie et recéption de SMSs
Envoie et recéption de SMSs
UTILISATEUR 1
UTILISATEUR 2
UTILISATEUR 1, 2. . .N

Architecture de l'application MobiTicket phases I et II.
Source : Braz, C. (CIRANO, 2003).

4.4. Spécifications non-fonctionnelles

Les spécifications non fonctionnelles sont toutes les spécifications qui n'expriment pas une fonction du logiciel. Ces spécifications expriment des contraintes : les contraintes d'interface et les contraintes de performance pour le MobiTicket phase I.

4.4.1. Contraintes d'interface

L'environnement de l'application: le programme doit s'exécuter sur le système d'exploitation Windows (Microsof Visual C++ 6.0). Le développement se fera en langage de programmation C++ lequel est une pré-condition pour l'utilisation del' AirCard® 555 dans un ordinateur portable.

L'environnement matériel : le programme doit s'exécuter sur un ordinateur portable en utilisant la carte réseau sans fil AirCard 555.

L'environnement humain : l'utilisateur devra gérer l'application en accomplissant les commandes suivants (Voir Figure 4) :

- Débuter et finaliser l'enchère VIP en utilisant les boutons : "Commencer l'enchère" et "Quitter".
- Vérification de l'arrivée et envoi des messages SMS en utilisant les boutons : "Récupérer de nouveaux SMSs".
- Saisir les numéros des téléphones mobiles dans la section "SMS des utilisateurs" sur les champs indiqués "À".

4.4.2. Contraintes de performance

Il s'agit dans ce cas notamment des contraintes de temps de réponse, c'est-à-dire, nous pouvons éprouver éventuellement des temps de réponse lent d'envoi ou de réception des messages SMS sur le téléphone mobile ou sur l'interface de l'application dû à instabilité des réseaux des communications sans fil ou parfois l'instabilité de ce type d'application. Cependant nous avons constaté pendant les tests du prototype que ces contraintes sont rares mais ils peuvent y arriver.

4.5. Information de maintenance

Le MobiTicket phase I pourra évidement se transformer dans une application de vocation commerciale. Cependant nous recommandons fortement l'exécution de la phase II avec le langage de programmation Java où l'intégration avec GNP2 serait moins coûteuse (pas besoin d'acheter l'outil JunC++ion) et plus simple.

5. Ergonomie du service

5.1. Introduction

L'ergonomie est définie comme "la science du travail". Elle s'utilise d'un ensemble de disciplines telles que médecine, physiologie, psychologie etc. afin d'adapter le travail et son environnement à l'homme. Pour parvenir à cet objectif, plusieurs facteurs sont à prendre en compte : physiologiques, sociaux, psychologiques. Cette ergonomie générale a ensuite été appliquée à différents domaines plus spécifiquement en notre cas l'informatique mobile.

Cette ergonomie au départ essentiellement physiologique a pour objectif le confort des opérateurs en adaptant le matériel à leurs postures. Par la suite, les ergonomes ont étudié les informations présentées à l'écran : couleurs, contrastes, formes des icônes. A partir de ces recherches, les ergonomes se sont intéressés aux Interactions Homme Machine et ont mis en évidence que :

- les difficultés rencontrées par les utilisateurs sont liées à la structure interne du logiciel (application) et non à l'apparence des écrans,
- les comportements et les habitudes des utilisateurs doivent être pris en compte au début de la conception afin d'optimiser l'utilisation d'un logiciel que ce soit pour un PC ou pour un périphérique mobile.

À l'heure actuelle, avec la croissance significative de l'Internet mobile, les connaissances de l'ergonomie sont précieuses pour le développement de services professionnels ou grand public pour ce marché porteur. L'ergonomie intégrée dès le début de la conception de l'application assure une communication efficace entre le client mobile et l'application.

L'ergonomie se mesure primordialement à partir de deux normes :

- la norme d'utilité qui correspond au besoin du client mobile,
- la norme "d'utilisabilité" qui correspond à la facilité d'utilisation.

5.2. Les offres des services MobiTicket

Le fournisseur de services d'application PEM (ASP) apportera à ses clients français et anglaises deux gamme de produits : Les ventes aux enchères des forfaits VIP (billets des spectacles couplés avec des offertes promotionnelles exclusives) qui nous appellerons dans ce document "enchères VIP" et les ventes de billets des spectacles invendus à la dernière minute qui nous appellerons "dernière minute" à partir de l'appareil sans fil des clients.

5.3. Les critères d'ergonomie du service

Nous avons réalisé assez des efforts, dans la conception de MobiTicket, d'obéir aux règles standard d'ergonomie d'un logiciel avec le but d'améliorer la communication utilisateur-appareil mobile, la facilité d'utilisation au même titre que l'on s'assure de la performance du service. En outre, il faudrait prendre en considération qu'une nouvelle application mobile n'est peut pas être adapté à la fois à tous les utilisateurs potentiels. Pour qu'elle soit adaptée à une population particulière, les caractéristiques de ces utilisateurs doivent être bien connues. Partant du constat que nous allons développer un prototype pour l'application et du fait que nous sommes limités par le temps, nous ne seront pas capable d'effectuer des testes d'usabilité et d'Interaction Homme-ordinateur afin d'effectivement identifier des problèmes d'utilisabilité auprès des utilisateurs réels (*end-users*), d'analyser les causes, de élaborer et de mettre en place des solutions dans la version suivante jusqu'à ce que les critères d'ergonomie soient validés. Ces testes pourront être réalisés dans un programme futur de travail.

À titre illustratif, nous nous utiliserons d'une grille qui décrira les différents éléments d'ergonomie et les attentes des clientes mobiles auprès de notre logiciel.

5.3.1. Critères ergonomiques d'utilisabilité du service

Un logiciel est utilisable lorsque le client mobile peut réaliser sa tâche (efficacité), qu'il consomme un minimum de ressources pour le faire (efficience) et que le système est agréable à utiliser (satisfaction). Le terme "Attentes" utilisé dans le tableau qui suit se réfère aux attentes des clientes mobiles auprès de notre logiciel.

Critères	Description	Attentes
Efficacité	*Concrétisation facile de l'achat des billets:* Le client mobile n'a qu'à se diriger au guichet du théâtre, présenter son code de confirmation et ensuite faire le paiement des billets.	Confiance et satisfaction vers le service.
	Accessibilité périodique au service : Disponible périodiquement, selon le profil du client mobile, les offres sont confortablement accessibles à partir des notifications des marchands participants (Push SMS).	Facilité/rapidité d'utilisation, déchargement de tâches et respect à la vie privée.
	Accessibilité au service en tout lieu : et de permettre tout à fait une accessibilité en tout temps et en tout lieu au client mobile.	Satisfaction de besoin remplie.
	Nombre minimum des messages envoyés côté client mobile : Dû à la difficulté de saisir des messages sur un petit clavier d'un appareil mobile, MobiTicket décharge le client mobile de la saisie des plusieurs messages afin d'effectuer sa transaction.	Confort d'utilisation et performance du service.

	Envoi unique d'un seul message SMS : Il faut seulement l'envoi d'un message SMS et aucun autre type de commande pour déclencher la transaction. Les détenteurs de téléphones mobiles savent bien comment faire pour envoyer un message SMS mais dans l'éventualité qu'ils ne savent pas, ces mêmes clients pourront trouver les instructions nécessaires pour la complétude de cette tâche sur le site Web du marchand.	Facilité d'apprentissage et mémorisation
Satisfaction	*Respect à la vie privé :* La stratégie Push SMS ne sera jamais invasive chez les clients mobiles puisque l'abonnement au service est une décision strictement du client mobile au moment qu'il rempli le formulaire d'inscription. En outre, le marchand en s'utilisant de l'outil de *profiling* proportionnera aux clients mobiles un service selon strictement ses préférences (*opt-in*).	Confiance et satisfaction vers le service.

5.3.2. Critères ergonomiques des interfaces du service

Dans cette section, notre objectif a été de rassembler une sélection de résultats ergonomiques aussi simplement et aussi clairement que possible.

Le terme "Attentes" utilisé dans le tableau qui suit se réfère aux attentes des clientes mobiles auprès de notre logiciel.

Critères	Description	Attentes
Compatibilité	*Respect au format des informations selon le contexte géographique :* Pour la version du logiciel en français, nous prendrons en compte le contexte géographique de la province du Québec tels que l'utilisation du signe de dollar après le chiffre (100$), le format de la date (2003.09.07) et de l'heure (24 heures).	Satisfaction et être à l'aise avec le service.
Concision	*Nombre réduit des interfaces :* Pour une transaction valide de l'enchère, que soit sans aucun type de message d'erreur, l'ergonomie du service s'utilise donc au total de seulement 4 écrans affichés au client mobile. Pour une transaction de dernière minute, l'ergonomie du service s'utilise donc au total de seulement 3 écrans affichés au client mobile.	Rapidité et performance du service.

Flexibilité	*Possibilité d'obtention de l'Aide :* Le client mobile obtient facilement d'aide à travers d'un numéro de téléphone gratuit (1-800) ou d'une interface Web qui contiendra toutes les informations nécessaires pour la réalisation de la transaction en incluant des FAQs. Par contre, nous avons donc limité le nombre d'écrans sur le téléphone mobile comme par exemple en n'offrant pas d'Aide directement dans l'application puisque diminuerait notamment la performance du service et augmenterait le nombre des erreurs chez les clients mobiles.	Confiance, intérêt et utilisation continue du service.
Complexité	*Nombre réduit d'options disponibles au client mobile dans la transaction :* Pour les enchères VIP : seulement une option de commande pour l'envoi de message SMS (mise). Pour le dernier minute : seulement une option de commande pour l'envoi du message SMS (consentement positif).	Facilité de compréhension et d'utilisation.
Feed-back et guidage	*Feed-back automatique aux demandes du client mobile :* La création d'un seul et unique "code d'accès abrégé" pour tous les opérateurs mobiles permettra la réponse automatique aux commandes de mise ou de consentement du client mobile en facilitant énormément sa tâche lorsqu'il n'aura pas besoin de taper le dit code pour l'envoi des ces commandes. En outre, l'adoption de ce "code d'accès abrégé" pour tous les opérateurs mobiles permettra une optimisation de tout le processus entre les principaux acteurs du service que soient le marchand, les opérateurs mobiles et les clients mobiles.	Satisfaction, facilité d'utilisation et confiance vers le service.

	Information rapide sur le succès ou l'échec des actions des clients mobiles : Après avoir fait une mise valide pour l'enchère VIP, le système (marchand) envoi un acquittement au client mobile en informant que la mise est valide et la transaction aura une continuité : *Merci! Votre mise a été acceptée! Attendez un message d'avis sur l'état d'avancement!* Après avoir consenti positivement pour la dernière minute, le système (marchand) envoi un acquittement au client mobile en informant qu'à ce moment-là il y en avait des billets disponibles, que ils ont été réservés et que la transaction a été complétée : *Les billets sont réservés! 3536 Daniel Bélanger 2 $9* *Valide jusqu'à 19h30 ET Salle Wilfrid-Pelletier* *175 St-Catherine Str. T: 8422112* Après avoir consenti positivement pour la dernière minute, le système (marchand) envoi un acquittement au client mobile en informant qu'à ce moment-là les billets ont été alors épuisés, et la transaction a été échoué (Voir Figure 3). *Désolé, les billets sont épuisés !*	Confiance et satisfaction vers le service.
Charge informationnelle	*Nombre adéquat de caractères sur les écrans :* Conscient de la difficulté de visualisation du contenu des écrans (messages lourdes) et de la limitation quant au nombre maximum de caractères de chaque message SMS (jusqu'à 160 caractères) des appareils GSM, l'accent sur l'ergonomie a donc été conçue avec des écrans les plus légers possibles. Il faut remarquer que l'utilisation de plus de texte sur les écrans de finalisation des transactions (le client mobile a gagné l'enchère ou il a obtenu de billets de dernier minute) a eu le but d'épargner au maximum le temps de consultation dans l'aide (téléphone ou site Web) et de permettre tout à fait une accessibilité au client mobile en tout lieu.	Confort visuel, facilité d'utilisation et accessibilité permanente au service.
Navigation	Permet une circulation très rapide en facilitant énormément l'accès aux services par le client mobile puisque c'est le marchand qui prend l'initiative d'offrir un service à travers la technologie Push SMS; un choix de conception du service qui permet au client mobile	

	d'optimiser son temps de communication et au Marchand de se procurer des revenues selon ses besoins spécifiques.	
Gestion des erreurs	Prévoit les erreurs humaines en affichant messages des erreurs tels que : *Désolé, vous avez déjà fait une mise!* *Désolé, vous n'avez pas fait une mise valide!* *Désolé, cette enchère est déjà terminée!* Par contre, si le client mobile a réalisé des saisies de textes non-valides alors nous aurons en moyenne un total de 5 écrans selon les captures d'écrans (Voir Figure 7).	Confiance et satisfaction vers le service.

6. Annexe

6.1. Code source complète de l'application MobiTicket version 1.0 :

```
/**
*
* Name : SMSSampleDlg.cpp
* Purpose : Main dialog implementation file
* History : created Oct 10, 2003
* Application: MobiTicket Version 1.0 (initial version)
* Project: Projet d'enchère mobile (PEM)
* Author: Christin Braz, Équipe PEM.
* Institution: CIRANO.
*
**/

#include "stdafx.h"
#include "SMSSample.h"
#include "SMSSampleDlg.h"
#include "SwiApiCm.h"
#include "SwiAPI1xRtt.h"
#include "SwiSms.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
SWI_STRUCT_SMS_Status g_SmsStatus; //global variable for this SWI_STRUCT_SMS_Status
SWI_STRUCT_SMS_SendStatus g_SmsSendStatus; //global variable for this
SWI_STRUCT_SMS_SendStatus
/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About
class CAboutDlg : public CDialog
{
public:
CAboutDlg();
// Dialog Data
//{{AFX_DATA(CAboutDlg)
enum { IDD = IDD_ABOUTBOX };
```

```
//}}AFX_DATA
// ClassWizard generated virtual function overrides
//{{AFX_VIRTUAL(CAboutDlg)
protected:
virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support
//}}AFX_VIRTUAL
// Implementation
protected:
//{{AFX_MSG(CAboutDlg)
//}}AFX_MSG
DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
//{{AFX_DATA_INIT(CAboutDlg)
//}}AFX_DATA_INIT
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
CDialog::DoDataExchange(pDX);
//{{AFX_DATA_MAP(CAboutDlg)
//}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
//{{AFX_MSG_MAP(CAboutDlg)
// No message handlers
//}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CSMSSampleDlg dialog
CSMSSampleDlg::CSMSSampleDlg(CWnd* pParent /*=NULL*/)
: CDialog(CSMSSampleDlg::IDD, pParent)
{
//{{AFX_DATA_INIT(CSMSSampleDlg)
//}}AFX_DATA_INIT
// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : ProcessError
//
// Description : Helper function to process error codes
//
// Arguments : code - error code
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void ProcessError(SWI_RCODE code)
{
CString errorString;
errorString.LoadString(code);
if (!errorString.GetLength() )
{
errorString.Format(_T("Internal error - error code: %d"), code);
}
MessageBox(NULL, errorString, _T("API Message"), MB_OK);
}
```

```
/////////////////////////////////////////////////////////////////////////////
//
// Function : IsEmailAddress
//
// Description : Helper function to check for e-mail address
//
// Arguments : szAddress - CString with address to check for
//
// Returns : BOOL (TRUE if correct e-mail address, FALSE otherwise
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
BOOL IsEmailAddress(CString szAddress)
{
if (-1 != szAddress.Find(_T('@'), 0))
{
return TRUE;
}
else
{
return FALSE;
}
}
void CSMSSampleDlg::DoDataExchange(CDataExchange* pDX)
{
CDialog::DoDataExchange(pDX);
//{{AFX_DATA_MAP(CSMSSampleDlg)
DDX_Control(pDX, IDC_COMPORT, m_ComPortCombo);
DDX_Control(pDX, IDC_SMSINBOXLIST, m_SmsInboxList);
DDX_Control(pDX, IDC_PRIORITYCOMBOBOX, m_PriorityCombo);
DDX_Control(pDX, IDC_CONNECTCOMBO, m_ConnectCombo);
//}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CSMSSampleDlg, CDialog)
//{{AFX_MSG_MAP(CSMSSampleDlg)
ON_WM_SYSCOMMAND()
ON_WM_PAINT()
ON_WM_QUERYDRAGICON()
ON_BN_CLICKED(IDC_CONNECT, OnConnect)
ON_BN_CLICKED(IDC_DELETESMS, OnDeleteSms)
ON_BN_CLICKED(IDC_SENDSMS, MyOnSendSms)
ON_BN_CLICKED(IDC_GETNEWSMS, OnGetSms)
ON_NOTIFY(NM_CLICK, IDC_SMSINBOXLIST, OnClickSmsinboxlist)
ON_BN_CLICKED(IDC_END, OnEnd)
ON_BN_CLICKED(IDC_DELETE_SMS, OnDeleteSms)
ON_CBN_SELCHANGE(IDC_CONNECTCOMBO, OnSelchangeConnectcombo)
//}}AFX_MSG_MAP
ON_MESSAGE (WM_NOTIFICATION, OnNotification)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CSMSSampleDlg message handlers
BOOL CSMSSampleDlg::OnInitDialog()
{
CDialog::OnInitDialog();
// Add "About..." menu item to system menu.
// IDM_ABOUTBOX must be in the system command range.
ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
ASSERT(IDM_ABOUTBOX < 0xF000);
CMenu* pSysMenu = GetSystemMenu(FALSE);
if (pSysMenu != NULL)
```

```
{
CString strAboutMenu;
strAboutMenu.LoadString(IDS_ABOUTBOX);
if (!strAboutMenu.IsEmpty())
{
pSysMenu->AppendMenu(MF_SEPARATOR);
pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
}
}
// Set the icon for this dialog. The framework does this automatically
// when the application's main window is not a dialog
SetIcon(m_hIcon, TRUE); // Set big icon
SetIcon(m_hIcon, FALSE); // Set small icon

// TODO: Add extra initialization here
// initialize main variables
m_bIsConnected = FALSE;
m_ConnectCombo.SetCurSel(0);
m_PriorityCombo.SetCurSel(0);
m_ComPortCombo.SetCurSel(0);
m_ComPortCombo.EnableWindow(FALSE);
// setup Sms list control columns
CString strLabel;
strLabel = _T("De");
m_SmsInboxList.InsertColumn(0, strLabel, LVCFMT_LEFT, 65);
strLabel = _T("Priorité");
m_SmsInboxList.InsertColumn(1, strLabel, LVCFMT_LEFT, 70);
strLabel = _T("Reçu");
m_SmsInboxList.InsertColumn(2, strLabel, LVCFMT_LEFT, 110);
m_SmsInboxList.SetExtendedStyle(LVS_EX_FULLROWSELECT);
GetDlgItem(IDC_COMPOSEMSGBODY ) ->SetWindowText(_T("Jean Leloup 10 oct. 03 Place des
Arts   20 h 2 billets mezzanine + backstage + souper. "));
GetDlgItem(IDC_SENDSMSTO_BIDMIN) ->SetWindowText(_T("300$"));
GetDlgItem(IDC_SENDSMSTO_CLOT) ->SetWindowText(_T("16h15"));
return TRUE; // return TRUE unless you set the focus to a control
}
void CSMSSampleDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
if ((nID & 0xFFF0) == IDM_ABOUTBOX)
{
CAboutDlg dlgAbout;
dlgAbout.DoModal();
}
else
{
CDialog::OnSysCommand(nID, lParam);
}
}
// If you add a minimize button to your dialog, you will need the code below
// to draw the icon. For MFC applications using the document/view model,
// this is automatically done for you by the framework.
void CSMSSampleDlg::OnPaint()
{
if (IsIconic())
{
CPaintDC dc(this); // device context for painting
SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);
// Center icon in client rectangle
int cxIcon = GetSystemMetrics(SM_CXICON);
int cyIcon = GetSystemMetrics(SM_CYICON);
CRect rect;
```

```
GetClientRect(&rect);
int x = (rect.Width() - cxIcon + 1) / 2;
int y = (rect.Height() - cyIcon + 1) / 2;
// Draw the icon
dc.DrawIcon(x, y, m_hIcon);
}
else
{
CDialog::OnPaint();
}
}
// The system calls this to obtain the cursor to display while the user drags
// the minimized window.
HCURSOR CSMSSampleDlg::OnQueryDragIcon()
{
return (HCURSOR) m_hIcon;
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : OnSelchangeConnectcombo
//
// Description : Connect Combo box changed
//
// Arguments : None
//
// Returns : None
//
// Comments : used to enable COM port selection when SB555 selected
//
////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::OnSelchangeConnectcombo()
{
if(3 == m_ConnectCombo.GetCurSel())
{
m_ComPortCombo.EnableWindow(TRUE);
}
else
{
m_ComPortCombo.EnableWindow(FALSE);
}
}

/////////////////////////////////////////////////////////////////////////////
//
// Function : OnConnect
//
// Description : Handler for connect control button
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::OnConnect()
{

SWI_TYPE_Device cardType;
if (!m_bIsConnected)
{
```

```
// AirCard 555 - 1xRtt
cardType = SWI_DEVICE_AC55x;

rCode = SwiApiOpen(cardType);

if (SWI_RCODE_OK == rCode)
{
GetDlgItem(IDC_CONNECT)->SetWindowText(_T("Arreter l'enchere"));
m_bIsConnected = TRUE;
my_bIsConnected = TRUE;
// register the callback function
rCode = SwiRegisterCallback(NotificationCallback);
if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
return;
}
// enable necessary SMS notifications
rCode = SwiNotify(SWI_NOTIFY_SmsStatus, DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
}
rCode = SwiNotify(SWI_NOTIFY_SmsSendStatus, DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
}
// enable other controls
GetDlgItem(IDC_SENDSMS)->EnableWindow(TRUE);
GetDlgItem(IDC_GETNEWSMS)->EnableWindow(TRUE);
GetDlgItem(IDC_DELETE_SMS)->EnableWindow(TRUE);
// initial check for new SMS
GetSms();
//delete all old SMS : CIRANO Auction
for(int i=0; i < m_File.iNumberOfEntries; i++)
{
m_File.deleteFromFile(i);
}
      //update the list box
      LoadSmsToListCtrl();
MessageBox("Debut de l'enchere");
// Modif for CIRANO Auction
OnSendSms("");
bid1=0; bid2=0;
CString s;
bidmin=0;
GetDlgItem(IDC_SENDSMSTO_BIDMIN)->GetWindowText(s);
for (int iter=0;s.GetAt(iter)==' ';iter++)
                        {;}
if (s.GetAt(iter)=='$')
                        {iter++;}
for (;s.GetAt(iter)==' ';iter++)
                        {;}
for (; iter < s.GetLength(); iter++)
                        {
                        switch (s.GetAt(iter))
                              {
                              case  '0' :
                              bidmin=(bidmin*10);
                              break;
```

```
                                    case  '1' :
                                    bidmin=(bidmin*10)+1;
                                    break;
                                    case  '2' :
                                    bidmin=(bidmin*10)+2;
                                    break;
                                    case  '3' :
                                    bidmin=(bidmin*10)+3;
                                    break;
                                    case  '4':
                                    bidmin=(bidmin*10)+4;
                                    break;
                                    case  '5':
                                    bidmin=(bidmin*10)+5;
                                    break;
                                    case  '6' :
                                    bidmin=(bidmin*10)+6;
                                    break;
                                    case  '7':
                                    bidmin=(bidmin*10)+7;
                                    break;
                                    case  '8':
                                    bidmin=(bidmin*10)+8;
                                    break;
                                    case  '9':
                                    bidmin=(bidmin*10)+9;
                                    break;
                                    default :
                                    iter = s.GetLength();
                                    break;
                                    }
                              }


}
else
{
ProcessError(rCode);
}
}
else
{
// Modif for CIRANO Auction
if (mesg1 !="")
            {OnSendTargetSms(mesg1,adr1);mesg1="";}
      if (mesg2 !="")
      {OnSendTargetSms(mesg1,adr2);mesg2="";}

my_bIsConnected=FALSE;
GetDlgItem(IDC_CONNECT)->SetWindowText(_T("Debuter l'enchere"));
CString messageG, messageP;
messageG="Vous avez gagne l'enchere! 8575 Jean Leloup";
messageP="Enchere Jean Leloup.Votre mise a ete insuffisante. La mise gagnante:";
```

```
if (bid2>bid1)
      {
      MessageBox("bid 1 est null, bid 2 non");
      if (bid1 !=0)
            {
            CString tmp;
            tmp.Format("%d",bid1);

            messageG+=tmp;

            //messageG+=CString(bid1);
            messageG+=" $. Valide jusqu'a 19h30 ET Salle Wilfrid-Pelletier 175 Rue St-
Catherine T:842-2112";
            messageP+=tmp;
            //messageP+=CString(bid1);
            messageP+="$";

            //OnSendTargetSms(messageP,num1);


            MessageBox(num1);
            tmp+="bid 1";
            //MessageBox(tmp);
            //OnSendTargetSms(messageG,num2);

            OnSendTargetTwoSms(messageP,num1,messageG,num2);
            MessageBox("call of Send two bid1!=0");
            }
      else
            {
            CString tmp;
            tmp.Format("%d",bid2);
            messageG+=tmp;
            //messageG+=CString(bid2);
            messageG+=" $. Valide jusqu'a 19h30 ET Salle Wilfrid-Pelletier. 175 Rue St-
Catherine. T:842-2112";
            tmp+="bid 2";
            //MessageBox(tmp);
            OnSendTargetSms(messageG,num2);


            }

      MessageBox(num2);


      }
else
      {

      if (bid2 !=0)
            {
            //MessageBox("else and bid2 different de zero ");
            CString tmp;
            tmp.Format("%d",bid2);
            messageG+=tmp;
            //messageG+=CString(bid2);
            messageG+=" $. Valide jusqu'a 19h30 ET Salle Wilfrid-Pelletier. 175 Rue St-
Catherine. T:842-2112";
            messageP+=tmp;
            //CString(bid2);
```

```
            messageP+="$";
            //OnSendTargetSms(messageP,num2);
            tmp+="bid 2";
            //MessageBox(tmp);
                  if (bid1 !=0)
            {
            //OnSendTargetSms(messageG,num1);
            OnSendTargetTwoSms(messageG,num1,messageP,num2);
            //MessageBox("call of Send two bid2!=0");
            }
                  else
                        OnSendTargetSms(messageP,num2);

            }
      else
            {
            //MessageBox("else and bid2 egale a  zero ");
            CString tmp;
            tmp.Format("%d",bid1);
            messageG+=tmp;
            //messageG+=CString(bid1);
            messageG+=" $. Valide jusqu'a 19h30 ET Salle Wilfrid-Pelletier. 175 Rue St-
Catherine. T:842-2112";
            tmp+="bid 1";
            //MessageBox(tmp);
            if (bid1 !=0)
            {
            OnSendTargetSms(messageG,num1);

            }



            }

      }
      bid1=0;bid2=0;
      //MessageBox(num1);MessageBox(num2);
      for(int n=0; n < m_File.iNumberOfEntries; n++)
      {
      m_File.deleteFromFile(n);
      }
            //update the list box
      LoadSmsToListCtrl();
      MessageBox("Fin de l'enchere");

/* disable notifications
rCode = SwiStopNotify(SWI_NOTIFY_SmsStatus, DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
      {
      ProcessError(rCode);
      }
rCode = SwiStopNotify(SWI_NOTIFY_SmsSendStatus, DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
      {
      ProcessError(rCode);
      }
// close the API
rCode = SwiApiClose();
if (SWI_RCODE_OK == rCode)
      {
```

```
      GetDlgItem(IDC_CONNECT)->SetWindowText(_T("Connect"));
      m_bIsConnected = FALSE;
      // disable other controls
      GetDlgItem(IDC_SENDSMS)->EnableWindow(FALSE);
      GetDlgItem(IDC_GETNEWSMS)->EnableWindow(FALSE);
      GetDlgItem(IDC_DELETE_SMS)->EnableWindow(FALSE);
      m_PriorityCombo.SetCurSel(0);
      CleanUpEditBoxes();
      }
else
      {
      ProcessError(rCode);
      }

*/
}
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : OnGetSms
//
// Description : Handler for Get New Sms control button
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::OnGetSms()
{
// call to a worker function
GetSms();
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : GetSms
//
// Description : Function to get received SMS
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::GetSms()
{
MessageBox("Recuperer SMSs");

SWI_STRUCT_SMS_Status sSmsStatus;
SWI_STRUCT_SMS_RetrieveSms sSms;
U32 msgCount;
CString szTemp;
TCHAR szMessage[SWI_MAX_BUFFER_SIZE];
TCHAR szTime[SWI_MAX_TIME_SIZE];
GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T(""));
// any new messages
sSmsStatus.sizeStruct = sizeof(SWI_STRUCT_SMS_Status);
```

```
rCode = SwiGetSMSMessageStatus(&sSmsStatus, DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
      {
      ProcessError(rCode);
      return;
      }
//Find out how many messages are stored in the modem
msgCount = sSmsStatus.cntRegularMsg + sSmsStatus.cntUrgentMsg + sSmsStatus.cntVoiceMsg;
if (!msgCount)
      {
      GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T("Pas de messages a lire"));
      LoadSmsToListCtrl();
      if (mesg1 !="")
            {OnSendTargetSms(mesg1,adr1);mesg1="";}
      if (mesg2 !="")
      {OnSendTargetSms(mesg2,adr2);mesg2="";}

      return;
      }
// allocate the buffer for the message
sSms.sizeStruct = sizeof(SWI_STRUCT_SMS_RetrieveSms);
sSms.sizeBuffer = BUFFER_SIZE;
sSms.pMessage = new U8[BUFFER_SIZE];
// retrieve the messages
TCHAR szAddress1[MAX_SMSI_TL_ADDRESS];
TCHAR szAddress2[MAX_SMSI_TL_ADDRESS];
TCHAR szTempTime[MAX_TIME_SIZE];
BOOL different=FALSE;
I32 i;
SWI_TYPE_SMS_Priority ePriority = SWI_SMS_PRIORITY_NORMAL;

int bid;
int iter ;
m_File.getInfo(0, szAddress1, &ePriority, szTempTime);
m_File.getInfo(1, szAddress2, &ePriority, szTempTime);
while (msgCount > 0)
      {
      rCode = SwiRetrieveSMSMessage(&sSms, DEFAULT_TIMEOUT);
      msgCount = sSms.cntRemainingSMS;
      GetDlgItem(IDC_NEWSMS_NOTIF)->EnableWindow(FALSE);
      if (SWI_RCODE_OK != rCode)
            {
            ProcessError(rCode);
            return;
            }
      //extract the time and the Message to character arrays
      GetTime(sSms, szTime);
      UpdateSmsOutbox(sSms.pMessage, szMessage);
      if (my_bIsConnected == FALSE)
            {
            if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Desole, cet enchere est deja terminee!";
            //OnSendTargetSms("Desole, cet enchere est deja
terminee!",sSms.sHeader.szAddress);
            }
                  else
```

```
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Desole, cet enchere est deja terminee!";
            }

            }
      else
            {
            for (i = 0; (i < 11) ; i++)
                  {
                  if (sSms.sHeader.szAddress[i] != (U8)szAddress1[i] )
                        different= TRUE;
                  }
            if (different==false)
                              {
                  if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Desole, vous avez deja fait une mise!";
            }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Desole, vous avez deja fait une mise!!";
            }

            //OnSendTargetSms("--Encheres VIP--  Desole, vous avez deja fait une
mise!",sSms.sHeader.szAddress);
            }


            else
                  {
                  for (i = 0; (i < 11); i++)
                        {
                        if (sSms.sHeader.szAddress[i] != (U8)szAddress2[i] )
                        different= TRUE;
                        }

            if (different==false)
{
if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Desole, cet enchere est deja terminee!";
            //OnSendTargetSms("--Encheres VIP--  Desole, cet enchere est deja
terminee!",sSms.sHeader.szAddress);
            }
                  else
```

```
                        {
            for (i = 0; (i < 12) ; i++)
                              {
                              adr2[i]=sSms.sHeader.szAddress[i];
                              }
            mesg2="Desole, cet enchere est deja terminee!";
            }
}

                  //OnSendTargetSms("--Encheres VIP--  Desole, vous avez deja fait une
mise!",sSms.sHeader.szAddress);
            else
                  {
                  // New bid to handle auction

                  CString s = CString (szMessage);

                  for (iter=0;s.GetAt(iter)==' ';iter++)
                        {;}
                  if (s.GetAt(iter)=='$')
                        {iter++;}
                  for (;s.GetAt(iter)==' ';iter++)
                        {;}
                  bid=0;
                  for (; iter < s.GetLength(); iter++)
                        {
                        switch (s.GetAt(iter))
                              {
                              case  '0' :
                              bid=(bid*10);
                              break;
                              case  '1' :
                              bid=(bid*10)+1;
                              break;
                              case  '2' :
                              bid=(bid*10)+2;
                              break;
                              case  '3' :
                              bid=(bid*10)+3;
                              break;
                              case  '4':
                              bid=(bid*10)+4;
                              break;
                              case  '5':
                              bid=(bid*10)+5;
                              break;
                              case  '6' :
                              bid=(bid*10)+6;
                              break;
                              case  '7':
                              bid=(bid*10)+7;
                              break;
                              case  '8':
                              bid=(bid*10)+8;
                              break;
                              case  '9':
                              bid=(bid*10)+9;
                              break;
                              default :
                              iter = s.GetLength();
                              break;
```

```
                              }
                        }

                  if (bid <= bidmin)
                        {

if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg1="Desole, vous n'avez pas  fait une mise valide!";
            //OnSendTargetSms("--Encheres VIP--  Desole, cet enchere est deja
terminee!",sSms.sHeader.szAddress);
            }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Desole, vous n'avez pas fait une mise valide!";
            }



                        //MessageBox("bid non valide");
                        //OnSendTargetSms("--Encheres VIP-- Desole, vous n'avez pas
fait une mise valide!",sSms.sHeader.szAddress);
                        }
                  else
                        {
                        if (bid1==0)
                              {
                              bid1=bid;
                              for (i = 0; (i < 12) ; i++)
                                    {
                                    num1[i]=sSms.sHeader.szAddress[i];
                                    }
                              //MessageBox("first if ");MessageBox(num1);
      MessageBox(num2);
                              }
                        else
                              {
                              bid2=bid;
                              for (i = 0; (i < 12) ; i++)
                                    {
                                    num2[i]=sSms.sHeader.szAddress[i];
                                    }
                              //MessageBox("first else ");MessageBox(num1);
      MessageBox(num2);
                              }
                        CString bids = CString (bid);
                        m_File.addToFile(sSms.sHeader.szAddress,
sSms.sHeader.ePriority,szTime, szMessage);
                        if (mesg1=="")
                        {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr1[i]=sSms.sHeader.szAddress[i];
```

```
                                    }
            mesg1="Merci! Votre mise a ete acceptee! Attendez un message d'avis sur
l'etat d'avancement!";
                  }
                  else
                              {
            for (i = 0; (i < 12) ; i++)
                                    {
                                    adr2[i]=sSms.sHeader.szAddress[i];
                                    }
            mesg2="Merci! Votre mise a ete acceptee! Attendez un message d'avis sur
l'etat d'avancement!";
            }

                        //OnSendTargetSms("Merci! Votre mise a ete acceptee! Veuillez
patienter et attendre une reponse! ",sSms.sHeader.szAddress);
                        }

                  }
            }
      }

      }


      if (m_SmsInboxList.GetItemCount())
      {
            // set the selected item in list control to have index 0 (arbitrary)
            //
            m_SmsInboxList.SetItemState(0, LVIS_SELECTED, LVIS_SELECTED);
                        }

      m_SmsInboxList.SetItemState(0, LVIS_SELECTED, LVIS_SELECTED);
      GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T("SMS a ete recupere avec
succes"));

      //update the list box
      LoadSmsToListCtrl();

      // free the memory
      delete [] sSms.pMessage;


      UpdateData(FALSE);



}
/////////////////////////////////////////////////////////////////////////////
//
// Function : GetTime
//
// Description : Function to retrive the Time and place it in a character array
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
```

```
void CSMSSampleDlg::GetTime(SWI_STRUCT_SMS_RetrieveSms sSms, TCHAR *szTime)
{
//Convert the Hex digits to a Character String
wsprintf(szTime, _T("%X/%X/200%X - %X:%X:%X"),
sSms.sHeader.ts_mon, sSms.sHeader.ts_day,
sSms.sHeader.ts_yr, sSms.sHeader.ts_hrs,
sSms.sHeader.ts_min, sSms.sHeader.ts_sec);
return;
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : OnDeleteSms
//
// Description : Handler for Delete Sms message in the file
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::OnDeleteSms()
{
// Find the selected row
POSITION iRow = m_SmsInboxList.GetFirstSelectedItemPosition();
//delete the message currently seleted from the File
if(!m_File.deleteFromFile((int)iRow-1))
MessageBox(_T("Could Not Delete the Entry"));
else
{

m_SmsInboxList.DeleteItem((int)iRow-1);
GetDlgItem(IDC_MESSAGEBODY)->SetWindowText(_T(""));
GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T("Entry Deleted"));
}
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : OnSendSms
//
// Description : Handler for Send Sms control button
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////

void CSMSSampleDlg::OnSendSms(CString message)
{
// TODO: Add your control notification handler code here
SWI_STRUCT_SMS_StoreSms sSms;
BOOL bInterimEmailSolution = TRUE; // for now this one is always on
CString szMsgBody;
CString szAddress;
CString szAddress2;
CString szTemp;
```

```
if (message == "")
{
GetDlgItem(IDC_COMPOSEMSGBODY)->GetWindowText(szMsgBody);
if (!szMsgBody.GetLength())
{
MessageBox("Cannot Send a Blank Message");
return;
}
szMsgBody+=" Mise minimale: ";
CString s;
GetDlgItem(IDC_SENDSMSTO_BIDMIN)->GetWindowText(s);
szMsgBody+=s;
szMsgBody+=" Cloture:";
GetDlgItem(IDC_SENDSMSTO_CLOT)->GetWindowText(s);
szMsgBody+=s;

}
else
{
szMsgBody=message;
}
if (!szMsgBody.GetLength())
{
MessageBox("Cannot Send a Blank Message");
return;
}
//check for message length
if (BUFFER_SIZE < szMsgBody.GetLength())
{
MessageBox(_T("Message cannot exceed 200 bytes"), NULL, MB_OK);
return;
}
// initialize SMS header
InitSmsHeader(&sSms.sSMSHeader);
// get TO address
GetDlgItem(IDC_SENDSMSTO)->GetWindowText(szAddress);

// customer1=szAddress;
if ( szAddress.GetLength() )
{
      // set destination address
      for (I32 i = 0; i < szAddress.GetLength(); i++)
      {
      sSms.sSMSHeader.szAddress[i] = (U8)szAddress.GetAt(i);
      }
      sSms.sSMSHeader.addrLen = szAddress.GetLength();
      sSms.sSMSHeader.addrDigitMode = 0;
      sSms.sSMSHeader.addrNumMode = 0;
      sSms.sSMSHeader.addrNumType = 0;
      sSms.sizeSMSBody = szMsgBody.GetLength();
      sSms.pMsgBody = (U8*)szMsgBody.GetBuffer(sSms.sizeSMSBody);
      // random MsgId in range 0-50 - these limits are arbitrary
      sSms.nSMSId = rand();
      if (sSms.nSMSId > 50)
      {
      sSms.nSMSId = sSms.nSMSId % 50;
      }
      // store the message to the modem
      rCode = SwiStoreSMSMessage(&sSms, DEFAULT_TIMEOUT);

      if (SWI_RCODE_OK != rCode)
```

```
	{
	ProcessError(rCode);
	return;
	}

}

///second message

SWI_STRUCT_SMS_StoreSms sSms2;

// initialize SMS header
InitSmsHeader(&sSms2.sSMSHeader);
// get TO address
GetDlgItem(IDC_SENDSMSTO_SND)->GetWindowText(szAddress2);
// customer2=szAddress;
if ( (!szAddress2.GetLength()) && (!szAddress.GetLength()) )
	{
	MessageBox(_T("Les adresses sont vides!"), NULL, MB_OK);
	return;
	}
if ( szAddress2.GetLength())
{
	// set destination address
for (I32 i = 0; i < szAddress2.GetLength(); i++)
{
sSms2.sSMSHeader.szAddress[i] = (U8)szAddress2.GetAt(i);
}
sSms2.sSMSHeader.addrLen = szAddress2.GetLength();
sSms2.sSMSHeader.addrDigitMode = 0;
sSms2.sSMSHeader.addrNumMode = 0;
sSms2.sSMSHeader.addrNumType = 0;
sSms2.sizeSMSBody = szMsgBody.GetLength();
sSms2.pMsgBody = (U8*)szMsgBody.GetBuffer(sSms2.sizeSMSBody);
// random MsgId in range 0-50 - these limits are arbitrary
sSms2.nSMSId = rand();
if (sSms2.nSMSId > 50)
{
sSms2.nSMSId = sSms2.nSMSId % 50;
}
// store the message to the modem
rCode = SwiStoreSMSMessage(&sSms2, DEFAULT_TIMEOUT);



if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
return;
}

}


// send the message
rCode = SwiSendSMS(DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
return;
}
```

```
GetDlgItem(IDC_SENDSMS)->EnableWindow(FALSE);
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : UpdateSmsOutbox
//
// Description : Function to parse the message body to the Sms message outbox
//
// Arguments : szInputString - message to parse (pointer to U8)
// szMessage - character array to return message in
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::UpdateSmsOutbox(U8* szInputString, TCHAR *szMessage)
{
U8* pMsg = szInputString;
CString szBody = _T("");
U32 counter = 0;
while (*pMsg != 0)
{
if (*pMsg == 13)
{
szBody += '\r';
szBody += '\n';
}
else if (*pMsg == 10)
{
szBody += '\r';
szBody += *pMsg;
}
else
{
szBody += *pMsg;
}
++pMsg;
if ( counter && !(counter % ONE_LINE_LENGTH) && (*(pMsg-1) != 13))
{
szBody += '\r';
szBody += '\n';
}
counter++;
}
_tcscpy(szMessage, szBody);
return;
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : InitSmsHeader
//
// Description : Function to initialize SMS header to be sent with the message
//
// Arguments : 1. pSmsHeader - pointer to SMS header
//
// Returns : None
//
// Comments :
//
```

```
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::InitSmsHeader(SWI_STRUCT_SMS_CdmaHeader *pSmsHeader)
{
pSmsHeader->eParamMask = (SWI_TYPE_SMS_ParamMask) (SWI_SMS_PARAMASK_BEARER_REPLY |
SWI_SMS_PARAMASK_ADDRESS |
SWI_SMS_PARAMASK_USER_DATA |
SWI_SMS_PARAMASK_PRIORITY |
SWI_SMS_PARAMASK_REPLY_OPT |
SWI_SMS_PARAMASK_ALERT |
SWI_SMS_PARAMASK_LANGUAGE |
SWI_SMS_PARAMASK_CALL_BACK);

pSmsHeader->addrDigitMode = 0;
pSmsHeader->addrNumMode = 0;
pSmsHeader->addrNumPlan = 0;
pSmsHeader->addrNumType = 0;
pSmsHeader->eMsgIdType = SWI_SMS_MSGIDTYPE_SUBMIT;
pSmsHeader->eMsgCategory = SWI_SMS_MSGCAT_POINT_TO_POINT;
pSmsHeader->eTeleservice = SWI_SMS_TELESRV_CELL_MESSAGING;
pSmsHeader->ePriority = SWI_SMS_PRIORITY_NORMAL;
pSmsHeader->ePrivacy = SWI_SMS_PRIVACY_NOT_RESTR;
pSmsHeader->eLanguage = SWI_SMS_LANGUAGE_ENGLISH;
pSmsHeader->user_ack_req = 1;
pSmsHeader->dak_req = 1;
pSmsHeader->user_ack_req = 0;
pSmsHeader->serviceCategory = 0;
pSmsHeader->deliveryAlert = 1;
pSmsHeader->tlAckReq = 1;
pSmsHeader->subLen = 0;
pSmsHeader->callBackLen = 0;
pSmsHeader->dataEncoding = 2;
//pSmsHeader->msgDispMode = 0;
// Fill in Callback #
CString szCB;
// Default callback # = our current MIN
rCode = SwiGetPhoneNumber(szCB.GetBuffer(100), 100, DEFAULT_TIMEOUT);
szCB.ReleaseBuffer();
if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
szCB = _T("");
return;
}

for (int iLen = 0; iLen < szCB.GetLength(); iLen++)
{
pSmsHeader->szCallBack[iLen] = (char)szCB.GetAt(iLen);
}
pSmsHeader->callBackLen = szCB.GetLength();
pSmsHeader->eMsgCategory = SWI_SMS_MSGCAT_POINT_TO_POINT;
pSmsHeader->ePriority = (SWI_TYPE_SMS_Priority)m_PriorityCombo.GetCurSel();
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : LoadSmsToListCtrl
//
// Description : Function to load all SMS messages to list control
//
// Arguments : None
//
// Returns : None
```

```
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::LoadSmsToListCtrl()
{
TCHAR szTempFrom[MAX_SMSI_TL_ADDRESS];
SWI_TYPE_SMS_Priority ePriority = SWI_SMS_PRIORITY_NORMAL;
TCHAR szTempTime[MAX_TIME_SIZE];

m_SmsInboxList.DeleteAllItems();
//fill the list with contents from file
//Note: SMS messages are not saved in the modem after they are retrieved from the modem
// so we save them to a file for future use
for(int i=0; i < m_File.iNumberOfEntries; i++)
{
//read one entry at a time
if(m_File.getInfo(i, szTempFrom, &ePriority, szTempTime))
{
//fill the List Box with the Entry information
m_SmsInboxList.InsertItem(i, szTempFrom);

switch(ePriority)
{
case SWI_SMS_PRIORITY_NORMAL:
m_SmsInboxList.SetItemText(i, 1, _T("Normal"));
break;
case SWI_SMS_PRIORITY_INTERACTIVE:
m_SmsInboxList.SetItemText(i, 1, _T("Interactive"));
break;
case SWI_SMS_PRIORITY_URGENT:
m_SmsInboxList.SetItemText(i, 1, _T("Urgent"));
break;
default:
m_SmsInboxList.SetItemText(i, 1, _T("Emergency"));
break;
}

m_SmsInboxList.SetItemText(i, 2, szTempTime);
}
}
}
/////////////////////////////////////////////////////////////////////////////
//
// Function : CleanUpEditBoxes
//
// Description : Helper function to reset edit boxes
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
/////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::CleanUpEditBoxes()
{
m_SmsInboxList.DeleteAllItems();
GetDlgItem(IDC_SENDSMSSTATUS)->SetWindowText(_T(""));
GetDlgItem(IDC_READSMSSTATUS)->SetWindowText(_T(""));
GetDlgItem(IDC_MESSAGEBODY)->SetWindowText(_T(""));
```

```
GetDlgItem(IDC_COMPOSEMSGBODY)->SetWindowText(_T(""));
GetDlgItem(IDC_SENDSMSTO)->SetWindowText(_T(""));
m_PriorityCombo.SetCurSel(0);
}
////////////////////////////////////////////////////////////////////////////
//
// Function : OnEnd()
//
// Description : Handler function for END control button
//
// Arguments : None
//
// Returns : None
//
// Comments :
//
////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::OnEnd()
{
//delete all old SMS : CIRANO Auction
for(int i=0; i < m_File.iNumberOfEntries; i++)
{
m_File.deleteFromFile(i);
}
// if API is opened, clean up correctly, close, and exit
//if (m_bIsConnected)
{
// disable notifications
rCode = SwiStopNotify(SWI_NOTIFY_SmsStatus, DEFAULT_TIMEOUT);
rCode = SwiStopNotify(SWI_NOTIFY_SmsSendStatus, DEFAULT_TIMEOUT);
// close the API
rCode = SwiApiClose();
}
CDialog::OnOK();
}
////////////////////////////////////////////////////////////////////////////
//
// Function : NotificationCallback
//
// Description : Callback that receives all event notifications
//
// Arguments : pNotify - notification variant
//
// Returns : None
//
// Comments :
//
////////////////////////////////////////////////////////////////////////////
void NotificationCallback(SwiNotifyVariant *pNotify)
{
SWI_TYPE_Notify eNotify = pNotify->eNotify;
//assign values with the global variables
g_SmsStatus = pNotify->_SmsStatus;
g_SmsSendStatus = pNotify->_SmsSendStatus;
//post a message to the main window
AfxGetMainWnd()->PostMessage(WM_NOTIFICATION, eNotify, NULL);
}
////////////////////////////////////////////////////////////////////////////
//
// Function : OnNotification
//
```

```
// Description : Handler for received notifications
//
// Arguments : wParam - holds the notification type
// lParam - holds the value
//
// Returns : None
//
// Comments :
//
///////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::OnNotification(WPARAM wParam, LPARAM lParam)
{
switch(wParam)
{
case SWI_NOTIFY_SmsStatus:
GetDlgItem(IDC_NEWSMS_NOTIF)->EnableWindow(TRUE);


// Traiter reception message Auction
// call to a worker function
GetSms();
break;
case SWI_NOTIFY_SmsSendStatus:
if (SWI_SMS_ERRORCLASS_NoError == g_SmsSendStatus.eErrorClass)
{
GetDlgItem(IDC_COMPOSEMSGBODY)->SetWindowText(_T(""));
MessageBox(_T("Message envoye."), NULL, MB_OK);
//GetDlgItem(IDC_SENDSMSSTATUS)->SetWindowText(_T("Message envoye"));
}
else
{
MessageBox(_T("Erreur: Pas capable d'envoyer le message."), NULL, MB_OK);
GetDlgItem(IDC_SENDSMSSTATUS)->SetWindowText(_T("Erreur: Pas capable d'envoyer le
message."));
}
GetDlgItem(IDC_SENDSMS)->EnableWindow(TRUE);
break;
default:
break;
}
}
///////////////////////////////////////////////////////////////////////////////
//
// Function : OnClickSmsinboxlist
//
// Description : When the SMS listbox is clicked
//
// Arguments : not used
//
// Returns : None
//
// Comments : this will update the message box with which ever row is selected
//
///////////////////////////////////////////////////////////////////////////////
void CSMSSampleDlg::OnClickSmsinboxlist(NMHDR* pNMHDR, LRESULT* pResult)
{

TCHAR szMessage[MAX_BUFFER_SIZE];
POSITION iRow = m_SmsInboxList.GetFirstSelectedItemPosition();
if(!m_bIsConnected)
return;
```

```
//retrieve the Message from the Selected Row
if(m_File.getMessage((int)iRow - 1, szMessage))
GetDlgItem(IDC_MESSAGEBODY)->SetWindowText(szMessage);

*pResult = 0;
}


void CSMSSampleDlg::MyOnSendSms()
{
OnSendSms("");
}


void CSMSSampleDlg::OnSendTargetSms(CString message, TCHAR *szTo)
{
            //MessageBox("Norramlement call of ");return ;

SWI_STRUCT_SMS_StoreSms sSms2;
BOOL bInterimEmailSolution = TRUE; // for now this one is always on
CString szMsgBody;
CString szAddress;
CString szTemp;
CString sTo = CString (szTo);

if (message == "")
{
GetDlgItem(IDC_COMPOSEMSGBODY)->GetWindowText(szMsgBody);
szMsgBody+=" Mise minimale: ";
CString s;
GetDlgItem(IDC_SENDSMSTO_BIDMIN)->GetWindowText(s);
szMsgBody+=s;
szMsgBody+=" Cloture:";
GetDlgItem(IDC_SENDSMSTO_CLOT)->GetWindowText(s);
szMsgBody+=s;

}
else
{
szMsgBody=message;
}
// TODO: Add your control notification handler code here
if (!szMsgBody.GetLength())
{
MessageBox("Pas capable d'envoyer un message vide!");
return;
}
//check for message length
if (BUFFER_SIZE < szMsgBody.GetLength())
{
MessageBox(_T("Message cannot exceed 200 bytes"), NULL, MB_OK);
return;
}



// initialize SMS header
InitSmsHeader(&sSms2.sSMSHeader);

// set TO address
```

```
for (I32 i = 0; i < sTo.GetLength(); i++)
{
sSms2.sSMSHeader.szAddress[i] = (U8)sTo.GetAt(i);
}
sSms2.sSMSHeader.addrLen = sTo.GetLength();
sSms2.sSMSHeader.addrDigitMode = 0;
sSms2.sSMSHeader.addrNumMode = 0;
sSms2.sSMSHeader.addrNumType = 0;

      sSms2.sizeSMSBody = szMsgBody.GetLength();
      sSms2.pMsgBody = (U8*)szMsgBody.GetBuffer(sSms2.sizeSMSBody);

      // random MsgId in range 0-50 - these limits are arbitrary
      sSms2.nSMSId = rand();
      if (sSms2.nSMSId > 50)
      {
            sSms2.nSMSId = sSms2.nSMSId % 50;
      }

      // store the message to the modem
      rCode = SwiStoreSMSMessage(&sSms2, DEFAULT_TIMEOUT);

      if (SWI_RCODE_OK != rCode)
      {
            ProcessError(rCode);
            return;
      }


      // send the message
      rCode = SwiSendSMS(DEFAULT_TIMEOUT*20);
      if (SWI_RCODE_OK != rCode)
      {
            ProcessError(rCode);
            return;
      }


      GetDlgItem(IDC_SENDSMS)->EnableWindow(FALSE);


}

void CSMSSampleDlg::OnSendTargetTwoSms(CString message1, TCHAR *szTo1, CString
message2, TCHAR *szTo2)
{
CString sTo1 = CString (szTo1);
CString sTo2 = CString (szTo2);

SWI_STRUCT_SMS_StoreSms sSms;
BOOL bInterimEmailSolution = TRUE; // for now this one is always on

if (!message1.GetLength())
{
MessageBox("Pas capable d'envoyer un message vide!");
return;
}

//check for message lengt
if (BUFFER_SIZE < message1.GetLength())
{
```

```
MessageBox(_T("Message cannot exceed 200 bytes"), NULL, MB_OK);
return;
}
// initialize SMS header
InitSmsHeader(&sSms.sSMSHeader);

for (I32 i = 0; i < sTo1.GetLength(); i++)
{
sSms.sSMSHeader.szAddress[i] = (U8)sTo1.GetAt(i);
}


      sSms.sSMSHeader.addrLen = sTo1.GetLength();
      sSms.sSMSHeader.addrDigitMode = 0;
      sSms.sSMSHeader.addrNumMode = 0;
      sSms.sSMSHeader.addrNumType = 0;
      sSms.sizeSMSBody = message1.GetLength();
      sSms.pMsgBody = (U8*)message1.GetBuffer(sSms.sizeSMSBody);
      // random MsgId in range 0-50 - these limits are arbitrary
      sSms.nSMSId = rand();
      if (sSms.nSMSId > 50)
      {
      sSms.nSMSId = sSms.nSMSId % 50;
      }
      // store the message to the modem
      rCode = SwiStoreSMSMessage(&sSms, DEFAULT_TIMEOUT);

      if (SWI_RCODE_OK != rCode)
      {
      ProcessError(rCode);
      return;
      }



///second message

SWI_STRUCT_SMS_StoreSms sSms2;

// initialize SMS header
InitSmsHeader(&sSms2.sSMSHeader);
// get TO address
if ( (!sTo2.GetLength()) && (!sTo1.GetLength()) )
      {
      MessageBox(_T("Les adresses sont vides!"), NULL, MB_OK);
      return;
      }
if (sTo2.GetLength())
{
      // set destination address
for (I32 i = 0; i < sTo2.GetLength(); i++)
{
sSms2.sSMSHeader.szAddress[i] = (U8) sTo2.GetAt(i);
}
sSms2.sSMSHeader.addrLen = sTo2.GetLength();
sSms2.sSMSHeader.addrDigitMode = 0;
sSms2.sSMSHeader.addrNumMode = 0;
sSms2.sSMSHeader.addrNumType = 0;
sSms2.sizeSMSBody = message2.GetLength();
sSms2.pMsgBody = (U8*)message2.GetBuffer(sSms2.sizeSMSBody);
// random MsgId in range 0-50 - these limits are arbitrary
```

```
sSms2.nSMSId = rand();
if (sSms2.nSMSId > 50)
{
sSms2.nSMSId = sSms2.nSMSId % 50;
}
// store the message to the modem
rCode = SwiStoreSMSMessage(&sSms2, DEFAULT_TIMEOUT);



if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
return;
}

}


// send the message
rCode = SwiSendSMS(DEFAULT_TIMEOUT);
if (SWI_RCODE_OK != rCode)
{
ProcessError(rCode);
return;
}

GetDlgItem(IDC_SENDSMS)->EnableWindow(FALSE);

}
```

4. La politique de modalités des services mobiles



Le volet technologique MobiTicket

Politique de modalités des services MobiTicket

1. Introduction

Les modalités des services MobiTicket[9] proposées dans ce document pourront être mises en place à travers de deux stratégies de fourniture des services distinctes comme suit :

- La première se réfère à la modalité de services utilisant le modèle "fournisseur d'applications hébergées" (Application Service Provider – ASP).
- La deuxième se réfère à la modalité de services employant des programmes de licences MobiTicket octroyés par CIRANO et/ou ses partenaires aux entreprises, c'est-à-dire, dans l'éventualité que CIRANO et/ou ses partenaires souhaitent vendre des licences du logiciel MobiTicket pour des clients-entreprises.

2. Modalité de Services ASP

Cette modalité de services s'utilise du modèle de prestation de services appelé "ASP". ASP signifie "Application Service Provider", autrement dit "fournisseur de services applicatifs", ces services étant délivrés via le réseau. L'ASP regroupe les fournisseurs de services (prestataires et hébergeurs) et le marché même que ces services représentent. Par extension, les "services ASP" désignent les applications que nous pouvons utiliser en ligne, en devenant donc, locataires de logiciel. L'entreprise accède via Internet, ou à travers un réseau privé, à l'application hébergée chez l'ASP.

Cette pratique évite à l'entreprise l'achat et l'installation du progiciel et réduit les coûts de maintenance. L'offre ASP de base comprend normalement le support technique, le stockage, la mise à jour des licences et la formation. Elle fonctionne en mode "one to many".

Le format ASP[10] permet aux compagnies des délais d'installation plus rapides, de la possibilité de se concentrer davantage sur leur cœur de son métier et de l'accès aux technologies de pointe.

L'ASP pourra être propriétaire des logiciels ou pourra les louer des éditeurs de logiciel. L'ASP présente l'avantage de donner une bonne visibilité financière car le logiciel est généralement commercialisé sous la forme d'un abonnement mensuel et par utilisateur.

---

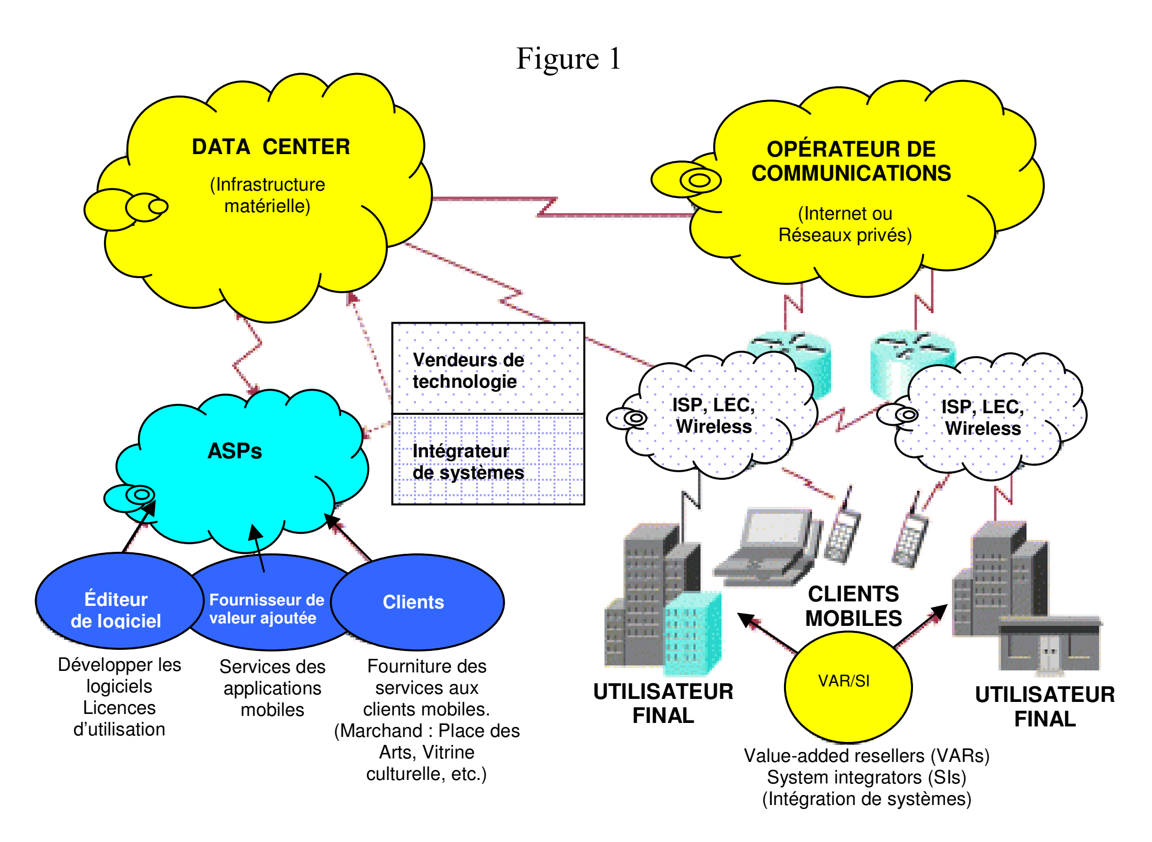
[9] Le terme "mobiTicket" n'est pas enregistré comme marque de commerce au Canada (Source: Industrie Canada, 2003).

[10] Les applications ASP génèreraient un retour sur investissement moyen de 404 % à 5 ans selon IDC (http://www.idc.fr/). Cet étude d'IDC intitulée "Impact financier des applications ASP", près de 50 % des sociétés ont amorti leurs dépenses en moins de six mois; 44 % des sociétés interrogées ont enregistré un retour sur investissement supérieur à 100 % et 12 % un retour sur investissement supérieur à 1 000 %. Et l'investissement moyen initial est de 399,000 dollars américaines.

Comme notre application MobiTicket touché notamment le domaine sans fil le modèle ASP serait mieux défini comme un Fournisseur des services applicatifs sans fil, que soit, un Wireless Application Service Provider (WASP).

Pour une meilleure compréhension de l'encadrement d'un fournisseur ASP dans la chaîne de distribution des services ASP nous présentons dans la Figure 1 les différents acteurs de ce segment de marché:

Figure 1



Écosystème d'un fournisseur d'applications.
Source : (Cisco Systems, Inc., 2002)

Selon la Figure 1, un utilisateur final ou un client mobile a besoin d'accéder à l'application MobiTicket qui est hébergée chez l'ASP (ou Data Center). En effet, le client mobile accède au site Web, par exemple, de la Place des Arts (marchand) où il pourra s'inscrire aux services de ventes aux enchères ou de dernière minute de billets des spectacles, et de cette façon recevoir des messages SMS du marchand correspondants aux services abonnés.

ÉDITEUR DE LOGICIEL

Ce sont des sociétés qui vendent des licences de logiciels au fournisseur ASP. L'éditeur garantit la fiabilité du fonctionnement de ses logiciels en mode client/serveur au sein de l'infrastructure du centre d'hébergement, dans le respect des règles de sécurité et de confidentialité.

DATA CENTER

C'est une société d'informatique qui acquiert des installations matérielles (serveurs, dispositifs de stockage et de sécurité des données), des logiciels de serveurs d'applications. Cette société garantit la sécurité des

applications et des données, la compatibilité entre matériels et entre logiciels, une disponibilité 24h/24, des temps de réponse courts, l'évolutivité de la bande passante et la mise à jour des matériels et applications.

OPÉRATEUR DE COMMUNICATIONS

Les télécommunications peuvent transiter par des lignes RNIS (Réseau Numérique à Intégration de Service Numéris), ADSL ou des lignes louées qui doivent offrir une bande passante suffisante pour exploiter confortablement les applications. Les liaisons les plus fiables sont constituées par les réseaux privés physiquement séparés du réseau Internet public.

FOURNISSEUR DE VALEUR AJOUTÉE

C'est un fournisseur de services qu'offre une valeur ajoutée aux services de base. Un service à valeur ajoutée ne peut pas être utilisé tout seul, par exemple, avec un autre service de base.
A service provider who offers services that add value to other (primitive) services. (A value added service cannot be used alone, i.e. with another primitive service.)

FOURNISSEUR DE SERVICES APPLICATIFS (ASP - Application Service Provider)

C'est l'intermédiaire entre éditeurs de logiciels, opérateur de communications, et le client. Le fournisseur ASP et le Data Center sont fréquemment réunis au sein d'une seule société.

CLIENT

C'est l'entreprise qui demande le service d'ASP, par exemple, Place des Arts ou Vitrine culturelle, qui peut offrir à sa clientèle (utilisateur mobile) de nouveaux services tels que les ventes aux enchères ou les ventes de dernière minute de billets des spectacles sur un téléphone mobile.

2.1. Description de la modalité de services ASP

Notre recommandation est la commercialisation des applications sous la forme d'un abonnement mensuel et par utilisateur selon la tarification des services présentée dans la section 2.2. ci-dessous.

2.2. Tarification des services ASP

Il est tout à fait important souligner qu'il n'y a pas un standard dans l'industrie d'ASP pour la tarification et pour l'utilisation de ces genres de services. Cependant nous indiquons dans le tableau 1 trois formats de tarification selon le groupe cible souhaité que pourront bien être employés pour la commercialisation des services :

Tableau 1 : Modèles de tarification

MODELES DE TARIFICATION	ELEMENTS DE LA TARIFICATION	PUBLIQUE CIBLE
1. Droits versés d'avance (upfront cost) plus mensuel	• Frais versés d'avance pour l'intégration, consultation et/ou pour la "customization" – option de transférer des logiciels et des matériels après un période déterminé. • Frais mensuels basés sur l'usage, sur le nombre d'utilisateurs spécifiques, ou sur le nombre d'utilisateurs concurrents.	Grandes compagnies avec des besoins d'implémentation plus complexes.
2. Forfaits mensuelles	• Droits versés d'avance (*upfront cost*) inclus dans les forfaits mensuels.	Compagnies PMEs avec des besoins mineures d'intégration et d'installation, ou compagnies *start-up*.
3. Distribution du revenu basée sur la flexibilité.	• Droits versés d'avance (*upfront*) mais pas de frais fixes mensuels – la tarification est basée sur le succès obtenu par l'application en incluant le nombre d'utilisateurs et le volume de transactions.	Dot com's deploiant sites Web de commerce électronique ou B2B.

Source: Current Analysis, 2000

Le modèle de tarification idéal pourrait mixer un ou plus modèles cités ci-dessus afin de capitaliser au maximum les revenues d'ASP. Cependant nous pouvons remarquer que les modèles 2 et 3 s'intègrent bien à l'industrie des spectacles.

3. Licences

La majorité des logiciels sont accompagnés d'un droit de licence accordé par l'éditeur. La licence donne le droit de copie, d'installation et d'utilisation du logiciel préalablement acquis pour un nombre d'utilisateurs ou postes de travail déterminé. Un document, appelé certificat de licence, matérialise ce droit.

Pour réduire le budget logiciel, les entreprises peuvent acquérir des différentes options de licences. Néanmoins chaque programme de licences comprend ses spécificités propres afin de répondre aux différentes attentes des clients. Ainsi, ces différents programmes peuvent proportionner que soit plus de souplesse, requérir un investissement initial important, répondre aux besoins multilingues, s'adapter aux processus de gestion d'une société utilisés pour acquérir et déployer des logiciels, ou de modifier complètement les procédures internes.

Les programmes de licences contribuent à réduire les coûts liés à l'achat des logiciels, à la gestion des licences et au développement de la technologie des clients. Pour cette raison il est vital de bien les planifier afin de déterminer quels programmes en matière de licences qui conviennent le mieux aux clients.

Le CIRANO peut être considéré comme un éditeur indépendant de logiciel en fournissant des licences pour les clients-entreprises.

3.1. Les Programmes de licences MobiTicket

Les options de licences pour le logiciel de services mobiles MobiTicket en comprenant les ventes aux enchères et les ventes de dernière minute de billets des spectacles prendra en compte initialement trois versions : Licence Transactionnelle MobiTicket, Licence Volume MobiTicket, et Licence Education MobiTicket.

Note : La structure tarifaire des licences, y compris les tranches de remises et les prix de chaque licence, sera considérée une activité future de travail.

3.1.1. Licence transactionnelle MobiTicket

Le programme de Licence Transactionnelle MobiTicket (LTM) est basée sur la transaction unique et recommandé pour les petites entreprises dans un seul pays. Cette licence permet aux clients de contrôler les dépenses en effectuant des achats individuels au fur et à mesure des besoins des utilisateurs.

Conditions d'achat :

Il n'y a pas de contrainte d'achat minimale, ni de volume. En outre, ce programme repose sur des certificats. Le client envoie un bon de commande et reçoit un certificat de licence électronique autorisant à effectuer une copie pour chaque licence achetée.

Caractéristiques :

- Il convient de remarquer que dans le cadre de la Licence Transactionnelle, chaque transaction est complètement indépendante. Cela permet de simplifier le processus d'achat pour les entreprises petites, tout en établissant un prix unique, aucun point n'étant associé aux transactions tels que la valeur minimale d'achat ou de volume.

- Assurance de mise a niveau[11] (optionnel) : Abonnement permettant de bénéficier de toutes les mises à jour de MobiTicket pendant une durée de 12 mois. Le renouvellement s'effectuera tous les 12 mois à la date anniversaire de la transaction. Il nécessite en principe d'avoir déjà la licence valide de MobiTicket que le client souhaite mettre à jour.

- Installation autorisée au bureau, sur le portable et à la maison (pas d'utilisation simultanée).

- Fourniture d'un certificat de licence avec les conditions générales de ventes.

- Manuel d'utilisation inclus.
- Inclusion de licence bilingue : Français et Anglais.
- Inclusion de licence multi-plateforme : peut être utilisé dans différents environnements (Dos, Windows, Mac OS…).

- CD-ROM (master[12]) d'installation du logiciel (optionnel).
- Fourniture de "documentations en ligne" (optionnel) qui peut substituer au manuel imprimé sur papier.

- Une seule clé de licence, quel que soit le nombre de logiciels.

---

[11] Assurance de mise a niveau permet à l'utilisateur ultime de garder le logiciel et les licences à jour avec la version la plus actuelle du logiciel. Le coût d'achat de l'assurance de mise à niveau est moins élevé que le coût d'achat des mises à niveau au fur et à mesure qu'elles paraissent, permettant à l'utilisateur ultime de prévoir le budget pour une période de un ou deux ans, puisque le coût est déterminé au moment de l'achat.
[12] Master : Support physique (disquettes ou CD ROM), sur lequel figure la version originale du logiciel qui va servir à l'installation du logiciel, généralement mentionné dans le certificat de licence.

3.1.2. Licence volume MobiTicket

Le programme de Licence Volume MobiTicket (LVM) est basée sur le volume et spécialement conçue pour les entreprises moyennes et grandes. Ce programme permet à l'entreprise d'acheter la licence du logiciel à un tarif contractuel.

Conditions d'achat :

- Commande initiale d'un montant minimum de 1 licence Serveur et 5 licences Poste de travail.
- Contrat de deux ans avec des options de renouvellement. Les renouvellements annuels sont obligatoires.

Caractéristiques :

- Un niveau de remise important (30% par rapport aux *produits pleins*).
- Une seule clé de licence, quel que soit le nombre de logiciels.
- Assurance de mise a niveau[13] : Abonnement permettant de bénéficier de toutes les mises à jour de MobiTicket pendant une durée de 12 mois. Il nécessite en principe d'avoir déjà la licence valide de MobiTicket que le client souhaite mettre à jour.
- Inclusion de version bilingue : Français et Anglais.
- Inclusion de licence multi-plateforme : peut être utilisé dans différents environnements (Dos, Windows, Mac OS…).
- Un an de Maintenance Gold inclus par chaque logiciel : inclut l'assurance mise a niveau, les mises à jour de sécurité et un support technique Gold pour le logiciel. Offerte pour une période d'un an, avec une extension possible d'un an supplémentaire.
- Fourniture de "documentations en ligne" (optionnel) qui peut substituer au manuel imprimé sur papier.
- Installation autorisée au bureau, sur le portable et à la maison (pas d'utilisation simultanée).

[13] Assurance de mise a niveau permet à l'utilisateur ultime de garder le logiciel et les licences à jour avec la version la plus actuelle du logiciel. Le coût d'achat de l'assurance de mise à niveau est moins élevé que le coût d'achat des mises à niveau au fur et à mesure qu'elles paraissent, permettant à l'utilisateur ultime de prévoir le budget pour une période de un ou deux ans, puisque le coût est déterminé au moment de l'achat.

3.1.3. Licence education MobiTicket

Le programme de Licence Education MobiTicket (LEM) est destiné strictement aux établissements d'enseignement, institutes techniques et centres de formation qui se bénéficient des conditions préférentielles d'acquisition (politique de prix à être établie).

Conditions d'achat :

- Achat d'un *produit plein* qui corresponde à une boîte contenant un certificat de licence d'utilisation pour un poste, le CD-ROM (master[14]) d'installation du logiciel et un manuel d'utilisation (Facultatif pour les nouvelles commandes).

- Une seule transaction, un seul achat. Il n'y a pas des quantités minimales d'achat. Le client pourra acquérir des licences en achetant 1 (une) licence pour un seul utilisateur, ou des licences pour des utilisateurs additionnels.

Caractéristiques:

- Assurance de mise a niveau[15] : Abonnement permettant de bénéficier de toutes les mises à jour de MobiTicket pendant une durée de 12 mois. Il nécessite en principe d'avoir déjà la licence valide de MobiTicket que le client souhaite mettre à jour.

- Inclusion de version bilingue : peut être utilisé dans deux langues : Français et Anglais.
- Inclusion de licence multi-plateforme : peut être utilisé dans différents environnements (Dos, Windows, Mac OS…).

- Une seule clé de licence, quel que soit le nombre de logiciels.
- Maintenance Gold (optionnel) : inclut l'assurance mise a niveau, les mises à jour de sécurité et un support technique Gold pour le logiciel. Offerte pour une période d'un an, avec une extension possible d'un an supplémentaire.

- Fourniture de "documentations en ligne" (optionnel) qui peut substituer au manuel imprimé sur papier.

- Installation autorisée au bureau, sur le portable et à la maison (pas d'utilisation simultanée).

---

[14] Master : Support physique (disquettes ou CD ROM), sur lequel figure la version originale du logiciel qui va servir à l'installation du logiciel, généralement mentionné dans le certificat de licence.

[15] Assurance de mise a niveau permet à l'utilisateur ultime de garder le logiciel et les licences à jour avec la version la plus actuelle du logiciel. Le coût d'achat de l'assurance de mise à niveau est moins élevé que le coût d'achat des mises à niveau au fur et à mesure qu'elles paraissent, permettant à l'utilisateur ultime de prévoir le budget pour une période de un ou deux ans, puisque le coût est déterminé au moment de l'achat.

5. Les règles d'enchères des services mobiles



Le volet technologique MobiTicket

Titre du document Instanciation Forfaits VIP	Référence FVIP001	**Version** **v3**	Date/Heure 09/09/2003 10:36	ID du client 001
Préparé par Christina Braz, Équipe PEM	Nom du fichier newEncheresForfaitsVIP.doc			Marchand Place des Arts

INSTANCIATION D'ENCHÈRE FORFAIT VIP

Type d'enchère	Second-Price Sealed-Bid
Type de service	Push SMS[16]
Description de l'enchère	Annonce d'une vente aux enchères VIP (billets + offres promotionnelles) par le marchand sur un téléphone mobile : *Jean Leloup 10 oct. 03* *Place des Arts 20 h* *2 billets mezzanine + backstage + souper* *300$ mise minimale* *Clôture: 16h15 ET*
Date de début	26.09.2003
Date de fin	26.09.2003
Nombre de Spectacles	1
Nombre des Phases	1
Nombre des Rounds	1
Règles de Négociation :	
Reference Side	SELL
Min_Active_Products	1
Min_Active_Parties	1
Ordre_Initial_Vente	300$[17]
Règles de Phase :	
Annonceur Adjudication Price Evolution	BACKWARD
Price Increment	$1[18]
Pricing Rule	Second_Price
Order Line Item Ranking	Price_Descending
	Time_Descending

[16] Service Push SMS: c'est un service mobile de télécommunications permettant de fournir, après une activation réussie du service par les clients mobiles eux-mêmes, des informations payantes sur un thème demandé au moyen de SMS. Les clients peuvent par exemple s'informer sur les actualités, les prévisions météorologiques, la bourse, etc.

[17] Prix de réserve : Ceci est inclus dans l'ordre initial d'annonceur.

[18] La moyenne d'incrément de prix d'une enchère est de 2 à 10%.

Phase End Conditions	Duration_Timeout = 450 secondes[19]
	Instant_Timeout = 04:15[20]
	Max_Rounds = 1
Round End Conditions	Duration_Timeout = 450 secondes
Règles de adjudication	Le client mobile[21] (CM) recevra des messages SMS étant dans l'un des cas suivants : 1a) Si le CM a fait une mise valide (SMS d'acquittement) : *Merci! Votre mise a été acceptée!* *Attendez un message d'avis sur l'état d'avancement!* 1a.1) Si le CM remporte l'enchère : *Vous avez gagné l'enchère! 8575 Jean Leloup 340$* *Valide jusqu'à 19h30 ET Salle Wilfrid-Pelletier* *175 Rue St-Catherine T: 842-2112* 1a.2) Si le CM perds l'enchère : *Enchère Jean Leloup. Votre mise a été insuffisante.* *La mise gagnante: 340$* 1b) Si le CM a misé plus d'une fois : *Désolé, vous avez déjà fait une mise!* 1c) Si le CM a misé une valeur inférieure ou égal à la mise minimale, ou a saisit des caractères autres que des lettres : *Désolé, vous n'avez pas fait une mise valide!* 1d) Si le CM a misé après l'heure de la fin d'enchère : *Désolé, cette enchère est déjà terminée!*

[19] La durée moyenne d'une enchère est de 450 secondes (15 minutes). Horaire : Entre 12h et 20h (weekdays) et entre 12h et 22h (weekends).
[20] Le moment exact de la fin de la phase.

Un document est créé par l'encanteur en faisant la mise à jour du l'état du marché (allocations finales) selon les informations ci-dessous. Ensuite, l'encanteur doit finalement communiquer aux participants le résultat de l'enchère (publication des adjudications).

`OnClosePhase()`[22]	Début de la production de l'adjudication
Cardinalité	1 par produit (spectacle) transigé
Acheteur	Client Mobile
Vendeur	Marchand
Produit	Spectacle Michel Rivard
Quantité	2 billets
Prix	Le prix gagnant (la plus baisse mise gagnante)
Acteurs autorisés à lire	Privée (les participants concernés)
Acteurs autorisés à modifier	Aucun : une fois sérialisée, l'adjudication ne peut être modifié.
Historique BD	Oui

Les adjudications produites dans `onClosePhase` seront par exemple: Une unité (2 billets) du Produit A (spectacle Michel Rivard) obtenu par le participant u4 pour $340.

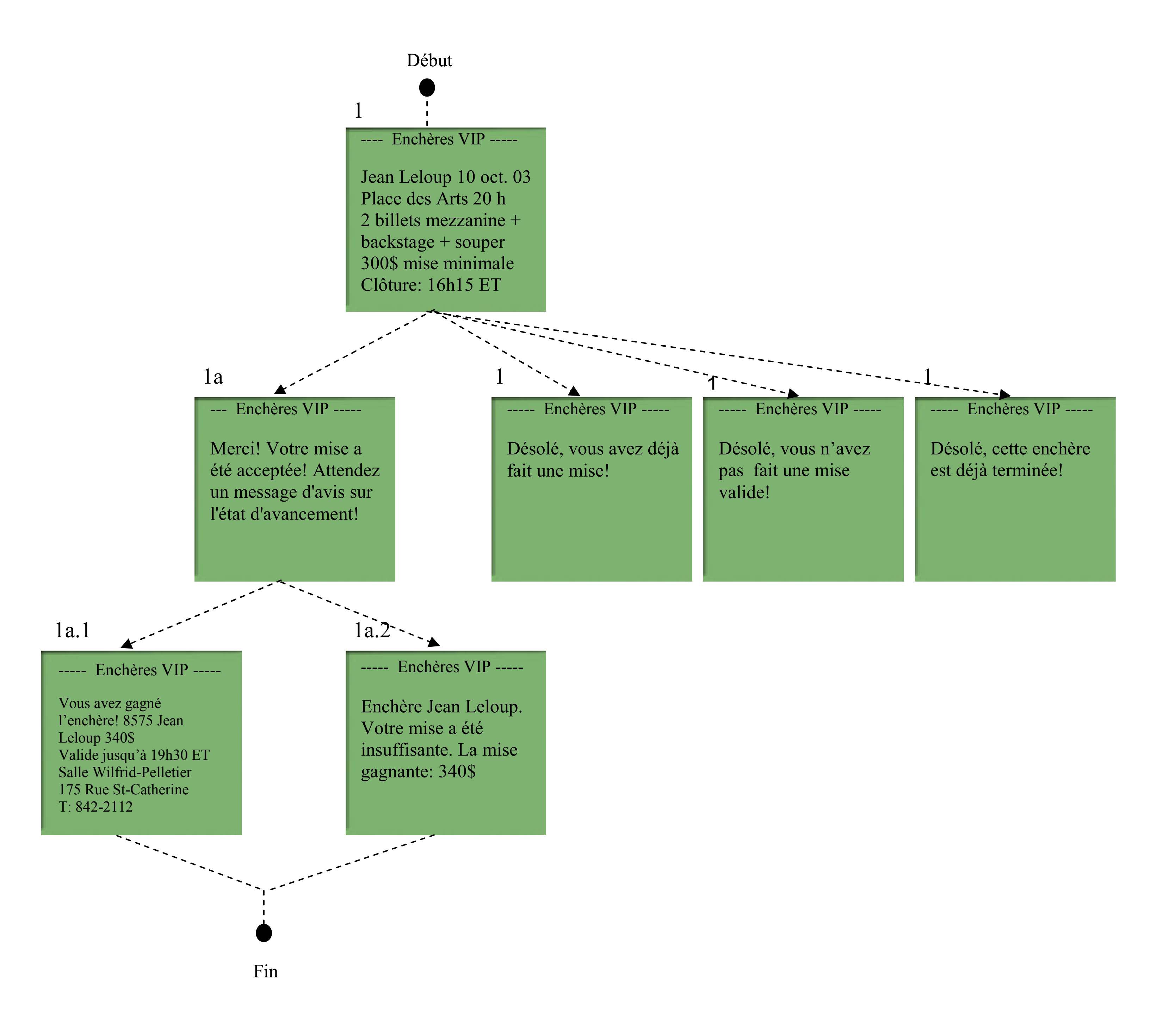
Ronde	Info observée		Mises			Allocations		
	Produit	Prix leader	Participant	Produit	Prix	Participant	Produit	Prix
1	A	340	u1	A	310	u1	A	310
			u2		305	u2		305
			u3		315	u3		315
			u4		340	u4		340
			u5		335	u5		335

Si deux ou plusieurs participants misent exactement les mêmes valeurs, l'encanteur considérera pour la définition du gagnant l'heure de création de l'`ordre` associé au participant.

Nombre de messages SMS à être envoyées au client mobile.	Défini par le profil du client mobile (*profiling*).

[21] Client Mobile : le détenteur d'un téléphone mobile en étant abonné au service d'un opérateur mobile.

[22] L'appel à la méthode `onClosePhase()`. L'encanteur y fait le traitement de fin de phase. C'est la dernière méthode à être appelée, mettant ainsi fin la boucle de l'encanteur.


Début
1
---- Enchères VIP -----
Jean Leloup 10 oct. 03
Place des Arts 20 h
2 billets mezzanine +
backstage + souper
300$ mise minimale
Clôture: 16h15 ET
1a
--- Enchères VIP -----
Merci! Votre mise a été acceptée! Attendez un message d'avis sur l'état d'avancement!
1
----- Enchères VIP -----
Désolé, vous avez déjà fait une mise!
1
----- Enchères VIP -----
Désolé, vous n'avez pas fait une mise valide!
1
----- Enchères VIP -----
Désolé, cette enchère est déjà terminée!
1a.1
----- Enchères VIP -----
Vous avez gagné l'enchère! 8575 Jean Leloup 340$
Valide jusqu'à 19h30 ET
Salle Wilfrid-Pelletier
175 Rue St-Catherine
T: 842-2112
1a.2
----- Enchères VIP -----
Enchère Jean Leloup. Votre mise a été insuffisante. La mise gagnante: 340$
Fin

Titre du document Instanciation DM	Référence IDM001	Version v3	Date/Heure 09/09/2003 10:36	ID du client 001
Préparé par Christina Braz, Équipe PEM		Nom du fichier newDernierMinute.doc		Marchand Place des Arts

INSTANCIATION D'OFFRE DE SERVICE "DERNIER MINUTE"

Description	Annonce d'une disponibilité des billets invendus de spectacles de "Dernier Minute (DM)" par le marchand sur un téléphone mobile : *Daniel Bélanger 23 oct. 03 8pm* *Place des Arts 2 billets parterre 100$* *Clôture: 16h15 ET*
Type de service	Push SMS[23]
Nombre de Spectacles	1
Nombre des Phases	1
Nombre des Rounds	1
Règles de Phase :	
Phase End Conditions	Duration_Timeout = 450 secondes[24]
	Instant_Timeout = 16:15[25]
Règles de Négociation :	
Reference Side	SELL
Min_Active_Products	1
Min_Active_Parties	1
Ordre_Initial_Vente	100$[26]
Règles de adjudication	Le client mobile[27] (CM) recevra des messages SMS étant dans l'un des cas suivants : 1a) Si le (a) CM emporte les billets : *Les billets ont été réservés! 3536 Daniel Bélanger 100$* *Valide jusqu'à 19h30 ET Salle Wilfrid-Pelletier* *175 Rue St-Catherine T: 842-2112*

[23] Push SMS : Par service "Push SMS", on entend un service mobile de télécommunications permettant de fournir, après une activation réussie du service par les clients mobiles eux-mêmes, des informations payantes sur un thème demandé au moyen de SMS. Les clients peuvent par exemple s'informer sur les spectacles disponibles, actualités, les prévisions météorologiques, la bourse, etc.

[24] La durée moyenne d'une enchère est de 450 secondes (15 minutes). Horaire : Entre 12h et 20h (weekdays) ; Entre 12h et 22h (weekends).

[25] Le moment exact de la fin de la phase.

[26] Prix de vente : Ceci est inclus dans l'ordre initial d'annonceur et défini par le marchand.

Note : 3536 Daniél Bélanger 100$ (Code du spectacle + Nom du spectacle + Valeur du billet).

1b) S'il n'y a plus de billets disponibles :

Désolé, les billets sont épuisés!

Un document est créé par l'encanteur en faisant la mise à jour du l'état du marché (allocations finales) selon les informations ci-dessous. Ensuite, l'encanteur doit finalement communiquer aux participants le résultat de l'enchère (publication des adjudications).

`OnClosePhase()`[28]	Début de la production de l'adjudication
Cardinalité	1 par produit (spectacle) transigé
Acheteur	Client Mobile
Vendeur	Marchand
Produit	Spectacle Daniel Bélanger
Quantité	2 billets
Consentement	Oui ou Non
Acteurs autorisés à lire	Privée (les participants concernés)
Acteurs autorisés à modifier	Aucun : une fois sérialisée, l'adjudication ne peut être modifié.
Historique BD	Oui

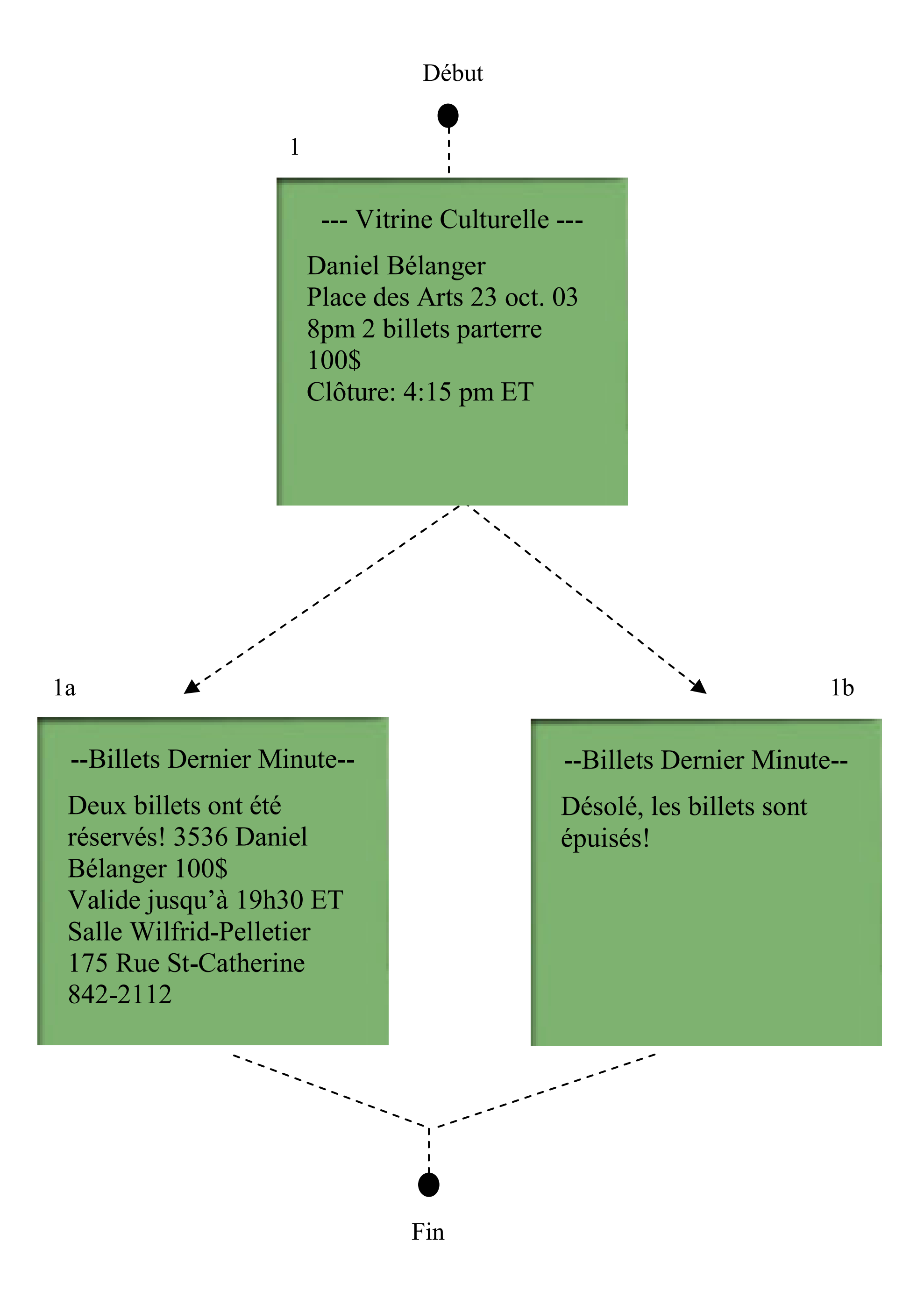
Les adjudications produites dans `onClosePhase` seront dans cet exemple :

Deux unités du Produit A (spectacle Daniel Bélanger) obtenu par les participants u1, u2, u3, u4 pour $100 pour une quantité totale offerte pour le marchand de 8 billets.

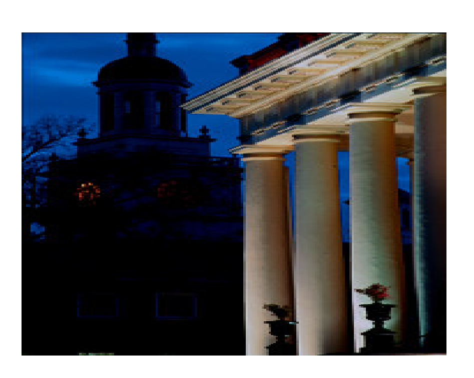
Ronde	Info observée		Mises				Allocations		
	Produit	Prix fixe	Participant	Produit	Cons.	Time-Stamp	Participant	Produit	Cons.
			u1			17:01:05	u1		Oui
1			u2			17:02:01	u2		
	A	100	u3	A	Oui	17:10:09	u3	A	
			u4			17:13:12	u4		
			u5			17:14:38	u5		

Nombre de messages SMS à être envoyées au client mobile.	Défini par le *profiling*.

[27] Client Mobile : le détenteur d'un téléphone mobile en étant abonné au service d'un opérateur mobile.
[28] L'appel à la méthode `onClosePhase()`. L'encanteur y fait le traitement de fin de phase. C'est la dernière méthode à être appelée, mettant ainsi fin la boucle de l'encanteur.


Début
1
--- Vitrine Culturelle ---
Daniel Bélanger
Place des Arts 23 oct. 03
8pm 2 billets parterre
100$
Clôture: 4:15 pm ET
1a
--Billets Dernier Minute--
Deux billets ont été réservés! 3536 Daniel Bélanger 100$
Valide jusqu'à 19h30 ET
Salle Wilfrid-Pelletier
175 Rue St-Catherine
842-2112
1b
--Billets Dernier Minute--
Désolé, les billets sont épuisés!
Fin

6. LES ASPECTS JURIDIQUES DES SERVICES MOBILES



LE VOLET TECHNOLOGIQUE
MOBITICKET

Politique d’utilisation des services mobiles
Aspects légaux

1. Introduction

La politique d’utilisation des services mobiles comprends deux axes distinctes : Une axe qui aborde le côté du fournisseur d’applications hébergées (Application Service Provider - ASP) et une autre qui aborde le côté du client mobile. Ce document a été structuré afin d’offrir une compréhension sur les procédures à être employés pour l’utilisation des services mobiles - aspects légaux - pour chaque axe.

Ces procédures couvrent la fourniture de services mobiles aux clients mobiles tels que :

- Des ventes aux enchères VIP des billets des spectacles sur un téléphone mobile.
- Des ventes à la dernière minute de billets invendus des spectacles sur un téléphone mobile.

L’ASP devra élaborer un Service Level Agreement (SLA) qui correspond à un accord entre un client-entreprise (marchand) et un fournisseur (ASP) sur le niveau de qualité de service offert par ce dernier. Le SLA est souvent suivi d'un Service Level Management (SLM) qui correspond à la Gestion de la qualité de service, lors de laquelle on s'assure que le fournisseur tient bien ses promesses d'un point de vue qualitatif, après avoir passé un SLA avec lui.

Notes : Dans ce document le masculine est entendu dans le sens générique et a pour but d’alléger le texte.

2. Définitions

"Service mobile" se réfère à la fourniture des moyens afin de permettre aux clients mobiles de participer aux ventes aux enchères VIP ou aux ventes de dernière minute des billets des spectacles sur un téléphone mobile.

"Téléphone mobile" se réfère à un appareil cellulaire digitale GSM connecté à un réseau GSM.

"SMS" (Short Messaging Service) se réfère à des messages comprenant des chiffres et/ou des textes contenant des informations sur les enchères et sur les mises liées aux services mobiles du marchand.
"Client mobile" se réfère au (à) détenteur(trice) d’un téléphone mobile en étant abonné au service d’un opérateur mobile.

3. L'axe Fournisseur d'applications hébergées (ASP)

L'ASP est ici indiqué comme une entreprise indépendante appelée "PEM-Projet d'enchère mobile".

3.1. Fourniture de services mobiles

- L'ASP s'utilisera dans la mesure du possible des tous les moyens raisonnables pour proportionner un service de qualité aux clients mobiles.

- Les Services mobiles sont fournis "tels quels" et "selon la disponibilité". ASP ne garantit pas un Service mobile en temps utile, sûr, sans erreur et ininterrompu, bien que ASP s'engage à faire effectivement des efforts afin d'offrir un service fiable aux clients mobiles.

- Le transport de messages SMS peut dépendre de facteurs en dehors du contrôle d'ASP en incluant, mais sans s'y limiter, aux facteurs affectant l'opération des opérateurs mobiles participants. L'ASP n'est pas obligée de fournir le Service mobile où tels facteurs puissent être prévenus. Pour éviter le soulèvement des doutes, un message SMS doit être considéré comme envoyé pour le client mobile dans la mesure qu'un signal d'acquittement aie été reçu par l'ASP pour cet effet.

- L'ASP est dans aucun cas responsable en matière de dommages, de blessures, et des réclamations des pertes du client mobile résultant de faille quelconque dans le transport ou dans le délai d'envoi des messages SMS quelconques au client mobile pour quelque que ce soit la raison en incluant, mais sans s'y limiter, aux des failles techniques du réseau ou des failles techniques de n'import quelle tierce partie dans laquelle ASP a la confiance en fournir le Service mobile.

- La priorité absolue d'ASP est d'assurer que les transactions des clients mobiles se déroulent avec succès. Néanmoins, il arrive parfois que le Service mobile d'ASP fasse l'expérience de pannes au cours desquelles les Clients mobiles ne peuvent pas enchérir dans les ventes aux enchères des VIP ou ne peuvent pas réserver des billets des spectacles à la dernière minute, ou être interrompu pour des motifs qui incluent, mais sans s'y limiter, les conditions environnementales, les limites techniques, les défauts ou lacunes, les limites des systèmes d'autres fournisseurs de services d'application mobile, les exigences d'urgence ou de sécurité publique, ou les causes indépendantes de la volonté d'ASP. Pour ces cas nous avons mis en place des clauses destinées à les traiter lorsque les pannes se produisent plus d'une heure mais moins de deux heures :

  (i) Sont inclus les objets dont l'enchérissement ou la réservation doit se terminer pendant la panne. Ceci concerne les fonctionnalités "Enchérir" ou "Réserver" des services mobiles.
  (ii) Pour la vente aux enchères des forfaits VIP, l'enchère sera prolongée d'accord avec le temps restant de clôture de l'enchère au moment de la panne. Une communication suivra d'ASP (message SMS) pour tous les Clients mobiles participants tout après que le système ait récupéré ses fonctions normales.

(iii) Pour la vente de billets des spectacles de dernière minute, la réservation (consentement positif) par le client mobile sera prolongée d'accord avec le temps restant de clôture de la vente au moment de la panne. Cependant, si la panne a été produite assez proche de l'heure de l'expiration de la validité des billets, le ASP se réserve le droit d'accorder les billets pour le client mobile qui ait fait le plus récent consentement au début de la panne. Le service d'horodatage ("Time-Stamping[29]") du système ASP assurera que cette règle sera attribuée au client mobile. Une communication suivra d'ASP (message SMS) pour tous les clients mobiles participants tout après que le système ait récupéré ses fonctions normales.
L'ASP se dégage de toute responsabilité liée aux garanties et aux modalités (expresses, implicites ou légales) en ce qui concerne le Service mobile et tout téléphone.

- L'ASP ne se responsabilise pas auprès du client mobile pour la violation des ses responsabilités liés aux délais ou aux *"non-performances"* qui sont dû à des événements quelconques en dehors du contrôle d'ASP.

- L'ASP s'engage à délivrer au client mobile des notifications des messages texte SMS et des confirmations des mises le plutôt possible, pourvu qu'elle reçoit ces notifications et confirmations du système du marchand concerné. Selon la charge de trafic du réseau, le client mobile pourra expérimenter des délais dans la réception de ses messages. Les notifications de fin-de-la-période d'enchère, *n'étant pas critique par rapport au temps*, sont sujettes à des délais. En outre, si le Téléphone mobile du client mobile est éteint ou hors de la zone de couverture dans le moment que l'ASP est en train de procéder l'envoi des messages, le client mobile ne les recevra pas jusqu'au moment que le Téléphone mobile soit allumé ou soit de retour à la zone de couverture.

[29] Time-Stamping : c'est l'enregistrement de l'heure de la création or existence de l'information.

4. L'axe Client mobile

- Avant d'utiliser les services mobiles d'ASP, le client mobile devra soigneusement lire les termes et conditions d'utilisation des services mobiles. Le client mobile sera légalement attaché à toutes ses engagements dans le moment qu'il clique sur le bouton "j'accepte ces termes et conditions", lequel nous considérons qui le client mobile les a bien lu, entendu et accepté.

- Le client reconnaît que le Service mobile peut être utilisé pour l'envoi des messages SMS pour les abonnées des opérateurs mobiles participants du Service mobile.

- Le client mobile doit reconnaître et être d'accord qui le transport de messages SMS peut dépendre de facteurs en dehors du contrôle d'ASP en incluant, mais sans s'y limiter, aux facteurs affectant l'opération des opérateurs mobiles participants. L'ASP n'est pas obligée de fournir le Service mobile où tels facteurs puissent être prévenus. Pour éviter le soulèvement des doutes, un message SMS doit être considéré comme envoyé pour le client mobile dans la mesure qu'un signal d'acquittement aie été reçu par l'ASP pour cet effet.

- Le client mobile ne pourra pas être chargé par son opérateur mobile participant dans la réception des messages d'erreurs provenant du système d'ASP.

- Le client mobile ne devra pas attendre des réponses d'ASP à tout et quelconque commande des messages SMS. Le client mobile seulement recevra des réponses des commandes "AIDE" et "STAT" si ce même client mobile ait s'inscrit pour tels services de commandes.

7. Bibliographie

Beck Zaratian, 2003 - *Microsoft Visual C++ 6.0 Programmer's Guide – The Essential Guide to Microsoft Visual C++ 6.0* - Microsoft Press.

Bérubé, Jean-François, 2003 – *GNP2 : Le guide du développeur version 2.0* – Centre interuniversitaire de recherche en analyse des organisations (CIRANO) - Professionnel de recherche.

Lipso Systems Inc., 2001 – *Where is Mobile Message headed?* – White-paper.

Microsoft Corporation, 2003 - *Microsoft Visual C++ 6.0 Deluxe Learning Edition* - Microsoft Press.

Application Service Provider (ASP), 2003.
http://www.mascre-heguy.com/htm/fr/conseils/conseil_asp_contrats_partenariat.htm

ASPNews.com 2003, *Top 20 Providers*, ASPNews.com Site Web.
http://www.aspnews.com/top20/article/0,,10341_753371_2,00.html

The ASPexchange.com, 2003 – Advisor for ASP Solutions.
http://www.theaspexchange.com/

ASPs: Netting it Out, 2003 - *ASPs must sell their value propositions to their customers*, by Geoff Dennis, ebizChronicle.com
http://www.ebizchronicle.com/spl_reports/november/asp_summit.htm

Bartolini, Claudio & Preist, Preist, 2001 - *A Framework for Automated Negotiation* – White Paper, Trusted E-Services Laboratory - HP Laboratories Bristol - HPL-2001-90.

Beam, Carrie/Segev, Ariev/Shanthikumar, George, 1996 - *Electronic Negotiation through Internet-based Auctions* CITM Working Paper 96-WP-1019 - Fisher Center for Information Technology & Management Walter A. Haas School of Business University of California, Berkeley Berkeley, CA 94720

Bérubé, Jean-François (professionnel de recherche, CIRANO) & Robert, Jacques (vice-président Groupe commerce électronique, CIRANO et professeur TI, HEC), 2001 - *Auction Rules Description*.
http://johnmolson.concordia.ca/gkersten/e_negotiations/Auction Rules Description.doc

eBid Ltd. 2003, *eBid.ie Web site.*
http://www.ebid.ie/eBidHelp/Help/eH?is=S#duration

Écho Gestion, 2003 - De l'ASP dans les cabinets d'expertise comptable.
http://pedagogie.ac-aix-marseille.fr/ecolyc/revue/N1/asp/

CIO.com 2003, *ASPs: The Next Chapter,* by Meridith Levinson – Sales and Marketing Automation.
http://www.cio.com/archive/050103/automation.html

Cisco Systems, Inc. 2002 - *Application Hosting Services: Opportunities for Service Providers* - Business Case.
http://www.cisco.com/warp/public/cc/so/cuso/sp/webhost/aphs_bc.htm

Codemesh Inc. 2003, *JunC++ion_ makes Java_ and C++ work together.*
http://isavix.net/whitepapers/1016810625589.pdf

CodeSegment Inc. 2003 - SMS Poll Center - Vojvode Stepe 139 11000 Belgrade Serbia & Montenegro
http://www.codesegment.com/

Corel Inc. 2003, Corel Licensing – *Invest in Maintenance* – Corel Web Site.
http://apps.corel.com/clp/maintenance.html
http://apps.corel.com/clp/transactional.html

Darwin 2003, *Executive Guides Application Service Provider (ASP)*, Darwin Site Web.
http://guide.darwinmag.com/technology/outsourcing/asp/

Electronic Business Group, 2003 **-** *Logiciel commercial, logiciel on line (ASP) ou logiciel libre ? -* Les dossiers du club – Le club des stratégies économiques.
http://www.ebg.net/dossiers/logiciels/3soluces.php

Ergonomie On line Web Site, 2002. Ce site présente un ensemble de liens vers des sites concernant l'ergonomie des IHM.
http://membres.lycos.fr/ergoline/

IBM Inc. 2003, Passport Advantage and Passport Advantage Express: Smarter, Faster, Easier
http://www-3.ibm.com/software/howtobuy/passportadvantage/

Mascré Heguy Associés, 2002 - *ASP : Quels contrats de partenariat ?* – Conseil du Mois Mascré Heguy Associés, société d'avocats.
http://www.bakom.ch/fr/telekommunikation/nummerierung/einzelnummerzuteilung/unterseite7/#sprungmarke1

Media Exchange 2003, *L'annonce des adjudications.*
http://www.mediaexchange.fr/cgv3102.html

Office Fédéral de la communication (OFCOM), 2003 - *Numéros de services à valeur ajoutée 0900, 0901, 0906 (SMS et numéros 090x)* Service de télécommunications.
http://www.bakom.ch/fr/telekommunikation/nummerierung/einzelnummerzuteilung/unterseite7/sprungmarke1

Place des Arts, 2003. *Place des Arts Site Web*.
http://www.pda.qc.ca

Progress Software Corp. 2003 - *The ASP Opportunity -* by Joe Alsop, President & Co-Founder, Progress Software Corp.
http://www.progress.com/partners/application/aspen/resources/executive/index.ssp

Sierra Wireless Inc., 2003, *Sierra Wireless Web Site.*
https://www.sierrawireless.com/

Symantec Inc. 2003, *Symantec Security License Program.*
http://www.symantec.com/region/fr/reseller/

Vachon, Julie/2003 - IFT6803 : Génie logicielle du commerce électronique (Chapitre 1 – Introduction, Processus de développement), Maîtrise en commerce électronique, Université de Montréal, Hiver 2003.
http://www.iro.umontreal.ca/~dift6803/Transparents/Chapitre1/chapitre1.1.pdf

VISUALTron Software Corporation, 2003 - *Wireless Short Message Services (SMS)* – Tutorials.
http://www.visualtron.com/wire_sms_topic04.htm

Xifin Company, 2003 - *Application Service Provider (ASP)*- Products and Services.
http://www.xifin.com/products/3PDTFrame.html?ASP.html~content

Zim Inc., 2003 - *Règles générales et définitions des services mobiles (Aspects juridiques)* - Commission de la capitale nationale (Ottawa).
http://ncc.zim.biz/nccupdater/legal_f.html

